A Heating Process Algorithm for Metal Forming by a Moving Heat Source

by

Hideki Shimizu

B.E., Naval Architecture, The University of Tokyo (1987)

Submitted to the Department of Ocean Engineering and the Department of Mechanical Engineering
in partial fulfillment of the requirements for the degrees of

Master of Science in Ocean Engineering

and

Master of Science in Mechanical Engineering

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

June 1997

Author .....
Department of Ocean Engineering
May 9, 1997

Certified by .....
Koichi Masubuchi
Professor, Department of Ocean Engineering
Thesis Supervisor

Certified by .....
Frank A. McClintock
Professor Emeritus, Department of Mechanical Engineering
Thesis Reader

Accepted by .....
J. Kim Vandiver
Chairman, Committee on Graduate Students
Department of Ocean Engineering

Accepted by .....
Ain A. Sonin
Chairman, Committee on Graduate Students
Department of Mechanical Engineering

A Heating Process Algorithm for Metal Forming by a Moving Heat Source

by

Hideki Shimizu

Submitted to the Department of Ocean Engineering and the Department of Mechanical Engineering
on May 9, 1997, in partial fulfillment of the
requirements for the degrees of
Master of Science in Ocean Engineering
and
Master of Science in Mechanical Engineering

Abstract

Metal forming by a moving heat source is a method to form a metal plate into a desired three dimensional shape for an external plate of ships, trains, and airplanes. A heat process determination algorithm is necessary for a proposed automated laser forming machine. Such an algorithm is developed here.

The entire process flow of laser forming was examined. As a result of the examination, it was realized that all past studies did not deal with the critical technology, the design algorithm for heat processes, that enables us to obtain a set of heat processes for a desired shape; all previous studies concentrated on the reverse problem of the heat process design algorithm that will predict the deformation for a certain heat process. Further review shows that computational analysis with the finite element method (FEM) is not accurate enough to predict general shapes that could be produced by heat processes. Moreover, computational analysis needs so much time to obtain a result that it cannot be used in designing heat processes.

Here, a possible design methodology is proposed. In the design methodology shown in this thesis, laser forming and line heating are considered as a forming method to produce a plastic strain distribution onto a plate in order to obtain a desired shape. The new methodology utilizes a Genetic Algorithm (GA) as an optimizing engine to determine the amount of heat input for each given heating line in order to obtain a desired shape. A set of heat process conditions is encoded into a string of binary bit for the use of a GA. Binary bit strings evolve over generation to generation following the natural selection scheme. Fitness values are calculated for their fitness indexes. Then, a decoding program that drives binary bit strings into physical shapes is required to calculate the fitness values. It decodes them to get an expected final shape by a set of heat process conditions. The decoded shape is compared to a target shape that is defined by a designer of products. The results of comparison are incorporated into a fitness value. The strings representing sets of heat process conditions evolve

opportunities to return a favor to him.

I would like to thank Ms. Muriel J. Bernier for her patient work to correct my English. She corrected not only my thesis but also my monthly reports on laser forming. She was a good teacher of English writing for me.

I thank Professor Patrikalakis, Dr. Maekawa, and Mr. Abrams at Design Laboratory. I feel a little sorry for Dr. Maekawa that I interrupted his research work many times with the demonstration of Praxiteles at the laboratory.

I also thank the members of laser forming project: Mr. Martukanitz and Mr. Sumnicht at Pennsylvania State University, Mr. Wooten at North American Division of Boeing Company, and Dr. Maher at Defense Division of Boeing Company. The discussions with them at Pennsylvania State University when we stayed there for the experiments were very good inputs for my research, and I could take a glance at the management style of American engineers. I hope that Mr. Martukanitz managed to come up with his Doctoral thesis this June.

I would like to thank Kawasaki Heavy Industries, Ltd. in Japan for giving me a wonderful opportunity to study at MIT. During my stay, I thought very much about what I should do in the company to make the company competitive in the global market. I could study not only engineering courses at the School of Engineering but also management courses at the Sloan School of Management. The extended one year given by the company was incredibly exciting for me since I could encounter many subjects at the business school that I have never known before, such as the Management of Technology and the Corporate Strategy.

My special thanks go to Dr. Susei, the senior executive of Kawasaki, because his decision brought me to MIT. If I had entered the other school that I originally intended to, I could not have learned many things and met these wonderful people. I really think that his broad vision was definitely right.

I cannot forget good discussions with Professor Ogawa at Kobe University and Mr. Inoue at Nippon Steel Corporation over the technology management. Professor Ogawa's patience gave me many insights into the subject. Mr. Inoue's ideas harmonized with mine.

My wife, Hiroko, was incredibly supportive during my stay at MIT. I could concentrate on my work at MIT since she did all the necessary chores so well, including the work related to the Japanese Association of MIT. I could not have accomplish my degree at MIT without her help.

I feel sorry that I cannot express my thanks to all of the people who helped me, encourage me, and enjoyed life with me. Otherwise, this acknowledgment will continue longer than my thesis itself. I would like to thank them all for giving me a wonderful time at MIT.

according to their fitness values to obtain the fittest heat process conditions for the target shape.

With the assumption that the additions of plastic strain distributions hold, i.e. no distortion of yield locus or Bauschinger effect, a heating condition determination algorithm has been implemented in the combination of a GA and a linear FEM program. The GA generates sets of process conditions repeatedly and the linear FEM program calculates expected shapes for given sets of conditions. The GA automatically searches the fittest shape to a target shape using the fitness value calculated with a fitness function. The linear FEM program has been validated with the comparison between calculated results and experimental results in terms of final shapes. Also, using the results of numerical calculation as target shapes, the GA obtained the correct set of process conditions. This validation shows the usefulness of the new methodology, if the plate is nearly flat during the process, for instance, within 10mm in a 300mm square plate 6mm thick. Calculation time of the program ranged from 10 to 50 hours on an HP 9000/715 workstation depending upon the complexity of the problem, i.e. the numbers of nodes, heating lines, and the number of possible heat intensities for each heating line.

Thesis Supervisor: Koichi Masubuchi
Title: Professor, Department of Ocean Engineering

Thesis Reader: Frank A. McClintock
Title: Professor Emeritus, Department of Mechanical Engineering

Acknowledgments

First of all, I would like to thank Professor Masubuchi. His kind and thoughtful support encouraged me to write this thesis. He is an excellent teacher in engineering fields, a considerate supporter for company matters, a helpful senior who helps an incompetent junior (of course it is me). He patiently taught me very much, not only engineering matters but also management and cultural things. All this support from him has added much to make my stay at MIT special for me. Of course, I learned many things in engineering from his engineering mind especially in welding, but also I learned many secrets or insights of life itself from the conversations with him over lunches. I cannot forget his favor to me for the rest of my life.

I also thank Professor McClintock for his thoughtful and careful instruction over my thesis. I guess that he struggled with my poor English writing ability. I have to say honestly that the original idea of using a Genetic Algorithm (GA) came from him. He kindly suggested that I attend a Ph.D. defense related to GA in the Department of Mechanical Engineering two years ago. If I had not attended the defense, I could not have come up with the application of GA to laser forming. His useful advice led me to the idea of the thesis.

I wish to thank Dr. Shibata at Artificial Intelligent Laboratory for instruction on the use of GA as an expert of GA. I could not have completed this thesis without his help for the development of the program.

I also thank Dr. Yang for the discussions on the system dynamic behavior of laser forming process. They give me deep understanding of a system dynamic aspect of laser forming.

Mr. Yung-Chin Hsiao at the Department of Ocean Engineering was my good research partner. We went to Pennsylvania State University many times to conduct experiments. He is also a good administrator of an HP workstation. I could not have carried out my thesis work without his help.

Mr. Li Liang at the Department of Ocean Engineering was my good lab mate. He is a warm hearted man. He helped me at many occasions. I regret to have less

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Background

Forming a three dimensional surface is essential for good fluid dynamic performance of transportation vehicles, such as ships, trains, airplanes, and space shuttles. Even though we have several techniques to produce three dimensional surfaces, the demands for achieving low production cost, less material deterioration, and high flexibility drives us to develop a new manufacturing technology to form them.

Die press forming commonly used in automobile and aerospace industries is suitable for high volume production. But it is not efficient for low volume production, since the dies are expensive to make. Also, it is not suitable for fabrication of three-dimensional parts in high strength materials and in large physical sizes, because a huge die press machine would be needed to produce those parts.

Line heating by a gas flame, which is mainly used in shipyards, is more versatile and useful than die forming to shape three dimensional shaped metal plates for external plates of ships and trains, since it does not need expensive dies and reshaping dies. However, line heating by a gas flame is difficult to automate in practical sites because using flammable gas is dangerous for operation without an operator, and a gas flame cannot give a constant heat flow onto a plate. Since achieving automation with a gas flame is difficult, material deterioration is a big issue for manual line heating with a gas flame.

Sometimes, aerospace companies machine a thin plate part from a chunk of material. Even though this seems to be prohibitive in terms of production cost, if we consider the initial cost of the required dies for production of the part and the material deterioration, machining the part is the right choice to produce the part.

1.2 Comparison among New Methods under Development

Many new methods to form three dimensional shape parts are under development to meet the demands mentioned above. They are discrete die forming, multi-point press forming, automated line heating with induction heating, and laser metal forming.

Discrete die forming, which adds flexibility to die forming, has been developed by Professor Hardt at MIT. The dies are divided into small columns which can slide back and forth in order to shape the top surface into a desired shape. If we put a flat plate between the two dies and press it with them, we can obtain a curved plate. Although a set of discrete dies can be used repeatedly for shaping various shapes, the discrete die forming machine will be large and expensive due to a huge force required to bind small columns into a die [54].

Multi-point press forming is similar to discrete die forming in that it forms a plate by physical restraints. Although it has been developed for more than twenty years in Japan, it still is not a practical method, since it requires a huge investment [36].

An automated line heating machine with induction heating coils has been under development in Japan. The characteristics of induction heating as a heat source are ideal for magnetic materials. It has good efficiency, consistent heat input onto a plate, and controllability of heat input. However, its usage is limited to magnetic materials because the heat is transferred to the product via electromagnetic waves [42, 43, 35].

Using a laser beam as a heat source is considered to be a good choice since it has good controllability of heat input onto a plate, and applies to many kinds of material, and a range of thickness. If we use a laser beam as a heat source, making

an automated line heating machine is simple because a laser beam is able to input a constant heat flux onto a metal plate, and controlling its output energy and movement on a plate is relatively easy. Nowadays, a high-power laser beam whose maximum output has reached 40 kW is available as a heat source and is ready for use in bending plates.

On the other hand, there are several drawbacks to using a laser beam as a heat source. The first is high cost for a machine due to the high price of a laser beam oscillator. However, the market price of a 3 kW-output CO_2 laser beam oscillator would be around $50,000 and it is still declining. We hope that the initial investment declines so much that it will soon be practical to construct an automated laser forming machine.

Secondly, the fluctuation of heat input by a laser beam due to changes in reflectivity is relatively large comparing to induction heating. In fact, the development team had to coat the specimens to obtain good experimental data. However, coating the material is prohibitive in practice. We will need to invent some closed loop algorithm to eliminate the effect of heat input fluctuation on the final shape of a plate.

Even though there are several disadvantages to using a laser beam, it is a good-quality heat source in terms of output stability and controllability. Also, its high output is attractive for metal forming.

1.3 Technical Issues in the Development of an Automated Laser Forming Facility

When we think of the development of an automated laser forming facility, there are many technical issues that we need to solve. Figure 1-1 describes an overall picture of the development subjects that we face. As shown in the figure, technical issues are classified into four categories.

1. Process Design Technique,

2. Material Database,

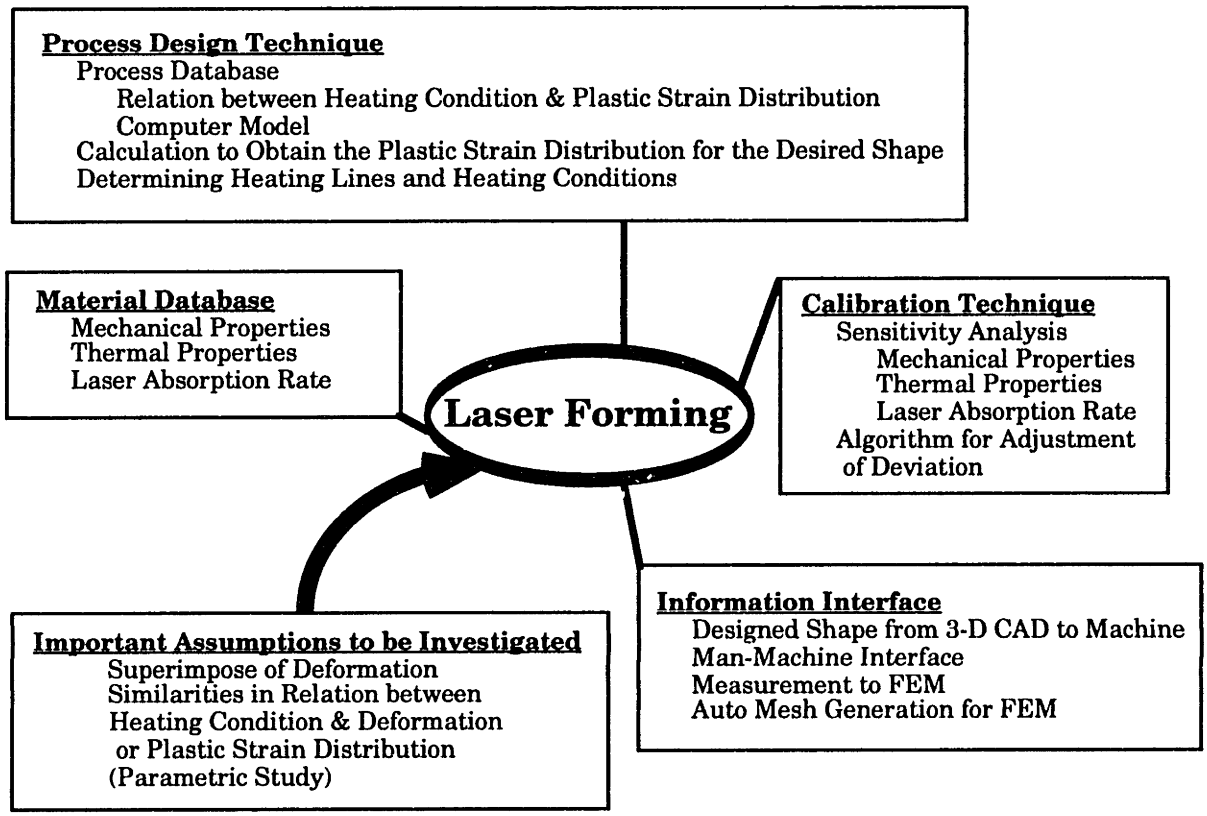


Figure 1-1: Key technical issues in the development of laser forming

3. Calibration Technique,

4. Information Interface.

Process Design Technique is the most important and difficult issue in the development. Although extensive researches have been done in establishing computer models using Finite Element Method (FEM), almost none of the researches were done in developing the heating condition determination procedure, as shown in the literature review in Chapter 2.

Material Database is needed to obtain a precise computer model. The material properties which are required for the calculation such as yield stress, tensile strength, and heat conductivity are strongly dependent on temperature. These properties are not available in many kinds of materials. We will need to do many measurements of these properties to obtain a precise database for FEM calculations.

Calibration Technique is required for laser forming since the discrepancy of the deformation of a plate is expected even if we design the heating process precisely with a complete set of material database. The discrepancy comes from the status of residual stress inside a plate, the surface finish, and the variation of material properties which cannot be controlled in an actual work site. We can speculate that operators of manual gas line heating can adjust the heating pattern or heat input naturally by experience. However, when we develop an automated forming machine, this will be a big issue.

Information interface is important when we develop a laser forming machine as a product. This subject is relatively easy, if we invest much budget for obtaining a good man-machine interface, with software technique used in mass production softwares.

After a glance of the entire picture and a review of related researches, I have decided to focus on the development of heating condition determination technique. The reason is that there is a lack of researches in this subject even though this is a critical issue for the development of an automated laser forming machine. Also, Professor Masubuchi, the supervisor of the thesis, has decided to focus on development of a precise computer model to reveal the mechanism of forming process and the

relationship between some principal parameter and deformation due to process. This part of the research has been carried out by Mr. Yung-Chin Hsiao.

1.4 Objectives of the Research

Since the development team is considering the application of laser forming to not only shipbuilding but also manufacturing aerospace products, the ultimate product of laser forming must be automated in order to minimize material damage. Almost all light-weight materials used for manufacturing aerospace products are strengthened by heat treatment. Therefore, we cannot melt a plate when we form the plate. If we traverse a heat torch manually, there is no guarantee to prevent material damage. We definitely need to develop an automated machine for the application of laser forming to heat sensitive materials.

After we set our ultimate goal of development of an automated laser forming facility, we can obtain two main objectives of the entire research:

1. Investigation of the interaction of a moving heat source and a plate,

2. Development of a design algorithm to form a plate into a desired shape.

Investigation of the interaction is related to establishing a precise computer model. We believe that automating a line heating process requires a deep understanding of thermo-elastic-plastic deformation in three dimensions in order to design heat processes correctly for a desired shape. This understanding has never been needed to conduct manual line heating by a gas flame because operators can control deformations and obtain desired shapes without a deep understanding of the complicated phenomenon. This belief led many researchers to the study concentrated on the analysis of thermo-elastic-plastic phenomena itself.

Recent researches on line heating or laser forming have been concentrated on the analysis by the FEM. If we use an FEM software, we can predict the deformation due to a certain heat process on a metal plate with a certain accuracy and also can understand the mechanism of a heat process.

However, the FEM software needs a long calculation time, for example forty hours for one heat process, to give a result because it needs a huge amount of calculation. In addition, the FEM software cannot tell us the heat conditions that are required to get a desired shape. As a result, we do not have any methodology to design heat processes to get a desired three dimensional shape in a metal plate. Therefore, the directions of the past researches seemed to be incorrect in terms of development of a heat process design methodology. A new paradigm of the development concept is needed to realize an automated laser forming machine.

The development of a design algorithm for a moving heat source is critical for the development of an automated laser forming facility. I have carried out this research activity under the supervision of Professor Masubuchi.

1.5 Scope of the Thesis

In Chapter 2, the literature review is presented in order to analyze the past studies on line heating or laser forming, to collect useful technologies that were examined in the studies, and to synthesize these technologies in the right direction for development of an automated laser forming machine. In the review, we did a literature search on investigations conducted in the past on line heating, laser forming, and computer simulations on line heating or laser forming as a database of state-of-the-art technologies for the laser forming development.

Chapter 3 presents the mechanism of a metal forming process by a moving heat source and the constitutive equations of the process. These understandings are helpful for predicting a final shape due to a certain heat process. However, the problem to design the heating processes is pointed out here. The mathematical understanding of the mechanism is not helpful for the heat process design process.

Then, in Chapter 4, a new design process of heating condition is proposed. Controllable parameters in heating processes are examined and the design procedure is simplified in order to make the problem solvable by state-of-the-art technologies. The problem is formulated into optimization and optimization methods are compared. Fi-

nally, the new heat process condition determination algorithm is presented with a GA.

Chapter 5 describes a program following the new methodology described in Chapter 4 to determine laser forming parameters for a pre-determined shape. The program utilizes a linear FEM program and a GA to determine the magnitudes of laser forming process for a given set of heating line positions.

The program has been validated with experimental results in Chapter 6. The validation of the program consists of two parts: the validation of the decoding program and the validation of the optimizing program. The validation of the decoding program is comparison of experimental final shapes from laser forming with the shapes calculated by the program. The validation of the optimizing program is the search results for a complicated shape whose heat process conditions are known. Also, a trial has been done using an experimental final shape from laser forming as a target shape in the validation of the optimizing program.

Chapter 7 presents the conclusion of the research and the future work.

Chapter 2

Literature Review

In this chapter, the researches conducted previously by many researchers are examined. The subjects covered here are line heating, numerical calculation of line heating, laser forming, and numerical calculation of laser forming.

2.1 Line Heating

Line heating is the most popular bending method in shipyards. Japanese shipyards started using this method in the 1950s [14], which means that line heating has a long history of about 40 years. Naturally, there are numerous technical reports on this method. However, these reports concentrated on the arrangement of experimental data using approximate theories [16, 1, 17, 52]. As a result, their conclusions are merely qualitative and cannot be utilized as a database for designing heat processes.

2.2 Numerical Calculation of Line Heating

After the Finite Element Method (FEM) and the Boundary Element Method (BEM) had been developed for better understanding of the physical phenomenon, many researchers have tried to simulate the phenomena of line heating. The objectives of these simulations by FEM or BEM were to obtain the distributions of stress and strain that are difficult to obtain in experiments, to understand the mechanism of

line heating, and to precisely predict the amount of deformation due to line heating; researchers tried to understand the physical phenomenon of line heating quantitatively rather than preceding researchers try to do qualitatively with approximate theories [31, 32, 34]. They obtained good agreements with the experimental results and the simulation results.

However, they achieved good agreements by adjustment of some parameters in the calculations after obtaining the experimental results. In other words, they still are not able to predict the deformation in advance. There were some reasons why they could not decide those parameters a priori: the difficulty of material modeling and the limitation of calculation speed of computers.

Theoretically speaking, if we use both a precise material model and an infinitesimal element size, we can obtain an accurate deformation. Nevertheless, it is impossible for us to model everything on non-linear material properties such as temperature dependent plasticity, strain rate dependent plasticity, temperature dependent thermal conductivity, etc.

Moreover, if we employ infinitesimal element size in calculation, FEM software cannot finish the calculation within a reasonable time period. This speed limitation prevented researchers from modeling the physical phenomenon precisely.

In order to overcome the limitation of calculation speed, Shin and Moshaiov tried to combine computer simulation method and an approximate theory. They tried to combine the computer simulation method with an approximate beam theory and to optimize the accuracy of a computer model [39, 40, 33]. Although they succeeded in obtaining a good agreement with experimental results in some extent, their method can be applied to only limited cases.

2.3 Computer Simulation of the Entire Process of Line Heating

Some researchers have tried to establish computer simulators for line heating using FEM or to establish the whole procedure of line heating. Nomoto et al. developed a simulator for line heating with FEM program. They only use elastic calculation to obtain thermo-elastic-plastic deformation for saving calculation time. They applied bending moments to a computer model of a plate to simulate heat processes. Their method had an advantage in reducing calculation time [37]. However, they paid little attention to accuracy of the result. That is why they mentioned that the simulator was developed for mainly educational use for line heating operators.

Ueda et al. have tried to establish the automated line heating method by use of FEM [51, 50, 46, 47, 44, 42, 48, 49, 45]. Their approach consists of three procedures: learning the line heating procedure from expert operators; developing FEM software that can simulate the mechanism of line heating; and establishment of shaping procedure using induction coil heaters. Although they have exhausted jobs on this subject, they could not establish the automated shaping procedure by using a moving heat source. Their method requires the manual inputs of heating positions and intensities of heat input by a software operator [42, 43, 35].

2.4 Laser Forming

The most detailed work on laser forming was conducted by Masubuchi et al. at MIT [28, 6, 18, 13, 29, 27, 3, 21, 22, 24, 23, 26, 38]. They conducted numerous experiments and searched for empirical parameters that constitute the deformation of laser forming. Also, their work showed that although a laser beam was a relatively stable heat source, the absorption rate varied by the condition of the plate surface and the amount of deformation which was affected.

Numerous simulations by FEM have been done to understand the mechanism of laser forming [8, 11, 19, 7, 9, 10, 20, 53]. These simulations were identical to

the simulations of line heating. The differences were spot size of a heat source and high density of heat flux of a heat source. In addition, most simulations of laser forming have been done by commercial software like ABAQUS, but, on the other hand, the simulations of line heating have been conducted with software developed by researchers themselves.

2.5 Evaluation of the Researches on Design Procedures of Heating Process Parameters

Past researches have been focused on the simulations of physical phenomenon of line heating or laser forming and the search for the parameters that constitute the result of deformation. A simulation by FEM allows us to predict the deformation of line heating or laser forming due to a certain heating condition with a certain amount of error.

However, the simulation by FEM is worthless in designing heat processes because the design process of heating conditions is the reverse problem of the simulation; the simulation can obtain a deformation when it is given a certain heating condition, but design process needs to produce a set of heating processes when a desired deformation is given. On the other hand, the search for the parameters is a good approach to make a database for designing heating processes. If we know the relationship between angular distortion and heating condition, we can produce desired curvatures in one direction by adding curvatures onto a plate linearly. Also, the relationship between shrinkage along a heating line and heating condition can be added to shape a plate.

The problem of this method is that no researchers have shown how to use the relationships in order to obtain a set of heating procedures for a desired shape. Ueda et al. have shown that the linearity of addition of deformation holds with an error of 15%. Also, they have shown that the linearity of addition of shrinkage holds with an error of 5% [42, 43]. These facts suggest that both deformation and shrinkage can be added to form a shape to design a set of heating processes for forming a three

dimensional shape. Therefore, in order to establish design methodology of heating processes, it is essential to complete a parametric survey that enables us to obtain a set of heating processes for a desired shape. After review of the previous studies, I have produced a design method to obtain a set of heating procedures for a certain desired shape.

2.6 Summary

The review of laser forming and other technologies related to laser forming shows that the simulation of thermo-elastic-plastic deformation is not beneficial for the establishment of the complete laser forming process. Because the simulation is a reverse process, it does not give us any set of heat procedures to form a desired shape. Also, the review shows that the parametric survey is useful but, in previous studies, it is not enough to establish the entire process of laser forming.

Chapter 3

Mechanism of Forming Process by a Moving Heat Source

3.1 Intuitive Understanding of the Mechanism

Forming process with a moving heat source is caused by residual strains produced along heating lines. When a heat source moves onto a metal plate, a heated-up portion of the plate will expand in all directions with constraints by the cool part around the heated up portion. Since the expansion is restricted, the residual strains are induced onto the plate. Depending on temperature distribution induced by a moving heat source, two kinds of residual strains will be induced in the plate:

1. Transverse Shrinkage,
2. Longitudinal Shrinkage.

The transverse shrinkage is perpendicular to the direction of movement of a heat source. If there is some difference between the amounts of transverse shrinkage on the top surface and on the bottom surface of the plate, the transverse shrinkage causes bending and produces angular distortion in the plate. (See Figure 3-1.) This is the case when we have shallow heat penetration by a moving heat source. The shallow heat penetration will be obtained with the combination of high traverse rate of the heat source and relatively small heat conductivity of the material.

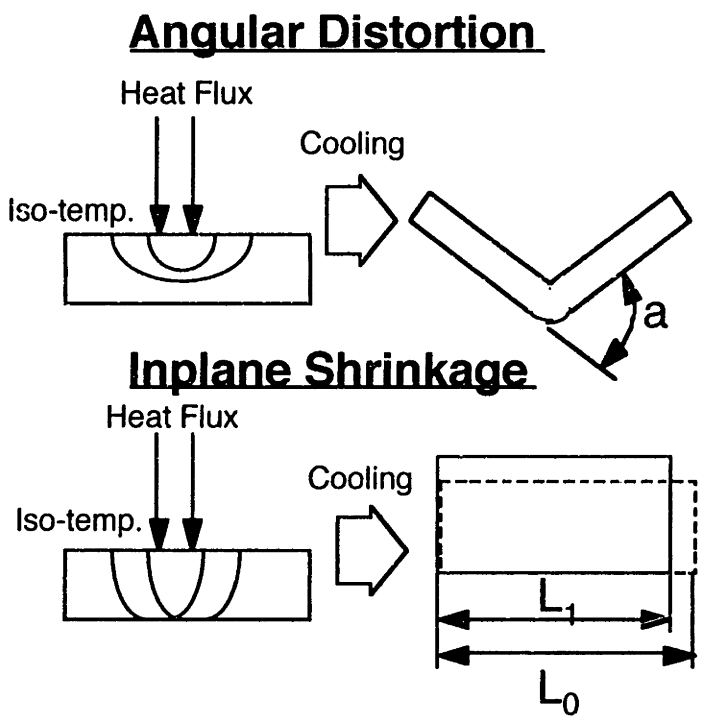


Figure 3-1: Deformations caused by a moving heat source

The longitudinal shrinkage parallel to the movement direction of the heat source is induced with the constraint in the longitudinal direction. It causes only inplane shrinkage when it is small as shown in Figure 3-1. However, when it is large and the stresses in the longitudinal direction are large enough to buckle the plate, it causes large out-plane deformation due to buckling.

In order to form a plate into a desired shape, we need to induce the residual strains onto a plate under the control.

3.2 Mathematical Description

As far as the material of a plate behaves elastically, the deformations can be expressed in mathematical formulations. In this section, the constitutive equations of the deformation of a plate due to a moving heat source are presented.

3.2.1 Mathematical Analysis of Heat Flow due to a Moving Heat Source

The following description on heat flow analysis is taken from the book written by Masubuchi [25].

If we use the following assumptions, we can analyze the temperature due to a moving heat source mathematically.

1. The heat source moves at a constant speed on or near the surface of a workpiece,
2. The size of the heat source is small compared to the size of the workpiece.

The fundamental equation of heat conduction in a solid is described in a cartesian coordinates as follows:

$$\rho c\frac{\partial\theta}{\partial t} = \dot{Q}_G + \frac{\partial}{\partial x}(\lambda\frac{\partial\theta}{\partial x}) + \frac{\partial}{\partial y}(\lambda\frac{\partial\theta}{\partial y}) + \frac{\partial}{\partial z}(\lambda\frac{\partial\theta}{\partial z}), \tag{3.1}$$

where ρ is density, c is specific heat, θ is temperature, t is time, λ is thermal conductivity, $\dot{Q}_G$ is the rate of temperature change due to heat generated per volume.

Equation 3.1 can be written,

$$\rho c\frac{\partial\theta}{\partial t} = \dot{Q}_G + \lambda[\frac{\partial^2\theta}{\partial x^2} + \frac{\partial^2\theta}{\partial y^2} + \frac{\partial^2\theta}{\partial z^2}] + \frac{\partial\lambda}{\partial\theta}[(\frac{\partial\theta}{\partial x})^2 + (\frac{\partial\theta}{\partial y})^2 + (\frac{\partial\theta}{\partial z})^2]. \tag{3.2}$$

If the value of thermal conductivity does not change with temperature, or $\partial\lambda/\partial\theta$=0, Equation 3.2 can be reduced to a linear differential equation:

$$\rho c\frac{\partial\theta}{\partial t} = \dot{Q}_G + \lambda[\frac{\partial^2\theta}{\partial x^2} + \frac{\partial^2\theta}{\partial y^2} + \frac{\partial^2\theta}{\partial z^2}]. \tag{3.3}$$

If there is no heat sink or source in the element concerned or $\dot{Q}_G = 0$, Equation 3.3 can ne further reduced to:

$$\frac{\partial\theta}{\partial t} = \kappa[\frac{\partial^2\theta}{\partial x^2} + \frac{\partial^2\theta}{\partial y^2} + \frac{\partial^2\theta}{\partial z^2}], \tag{3.4}$$

where $\kappa = \frac{\lambda}{c\rho}$ is thermal diffusivity.

In order to avoid mathematical complexity in solving a non-linear equation, Equation 3.2, we need to assume that thermal properties do not change with the change in temperature, i.e. $\partial\lambda/\partial\theta$=0.

Using a moving cartesian coordinate w,y,and z which moves at the same speed, v, as a heat source.

$$w = x - vt, \tag{3.5}$$

$$\frac{\partial w}{\partial x} = 1; \quad \frac{\partial w}{\partial t} = -v, \tag{3.6}$$

$$\frac{\partial \theta}{\partial x} = \frac{\partial \theta}{\partial w}\frac{\partial w}{\partial x}; \quad \frac{\partial^2 \theta}{\partial x^2} = \frac{\partial^2 \theta}{\partial w^2}. \tag{3.7}$$

The relationship between $\partial\theta/\partial t$ for the fixed coordinate $(\partial\theta/\partial t)_{FC}$ and that of the moving coordinate $(\partial\theta/\partial t)_{MC}$ is:

$$(\frac{\partial \theta}{\partial t})_{FC} = (\frac{\partial \theta}{\partial t})_{MC} + \frac{\partial \theta}{\partial w}\frac{\partial w}{\partial t} = (\frac{\partial \theta}{\partial t})_{MC} - v(\frac{\partial \theta}{\partial w}). \tag{3.8}$$

On the quasi-stationary state,

$$(\frac{\partial \theta}{\partial t})_{MC} = 0. \tag{3.9}$$

Then, Equation 3.4 can be expressed by the following equation:

$$\frac{\partial^2 \theta}{\partial w^2} + \frac{\partial^2 \theta}{\partial y^2} + \frac{\partial^2 \theta}{\partial z^2} = -\frac{v}{\kappa}(\frac{\partial \theta}{\partial w}). \tag{3.10}$$

Now, θ is a function of position (w, y, z) only.

Equation 3.10 is more easily handled by replacing it with the following expression:

$$\theta = \theta_0 + e^{-\frac{v}{2\kappa}w}\phi(w, y, z), \tag{3.11}$$

where $theta_0$ is the initial temperature and $\phi(w, y, z)$ is the function to be found.

Putting Equation 3.11 in Equation 3.10 and performing the calculations, we find

$$\frac{\partial^2 \phi}{\partial w^2} + \frac{\partial^2 \phi}{\partial y^2} + \frac{\partial^2 \phi}{\partial z^2} - (\frac{v}{2\kappa})^2\phi = 0. \tag{3.12}$$

We can solve the above differential equation mathematically with proper boundary conditions. For example, if we consider a three-dimensional case with finite thickness, we can use the following boundary conditions to solve the equation.

$$\frac{\partial \theta}{\partial z} = 0; \quad for \;\; z = 0 \;\; and \;\; z = h, \tag{3.13}$$

where h is thickness.

The solution is as follows:

$$\theta - \theta_0 = \frac{Q}{2\pi\lambda} e^{-(\frac{v}{2\kappa})w} [\frac{e^{-(\frac{v}{2\kappa})R}}{R} + \sum_{n=1}^{\infty} (\frac{e^{-(\frac{v}{2\kappa})R_n}}{R_n} + \frac{e^{-(\frac{v}{2\kappa})R'_n}}{R'_n})], \tag{3.14}$$

where,

$$R = \sqrt{w^2 + y^2 + z^2}, \tag{3.15}$$

$$R_n = \sqrt{w^2 + y^2 + (2nh - z)^2}, \tag{3.16}$$

$$R'_n = \sqrt{w^2 + y^2 + (2nh + z)^2}. \tag{3.17}$$

Equation 3.14 shows the temperature distribution at a given time due to a point moving heat source in quasi-stationary state. The temperature distribution expressed by the equation is the source of deformation. Since the temperature of heat is so concentrated that the heated portion cannot expand freely and the yield criterion is lowered by high temperature, plastic strain is induced into the plate.

3.2.2 Thermo-Elastic Problem

The following mathematical description on thermo-elastic problem is obtained from the book written by Boley and Weiner [4].

If we assume that the plate is isotropic, with the following principal assumptions used, we can describe the mechanism of the deformation mechanism well due to a moving heat source.

1. The temperature can be determined independently of the deformation of the body,

2. The deformations are small,

3. The material behaves elastically at all times.

The first assumption is required in order to omit the mechanical coupling terms in the heat condition equation. The second assumption defines that the displacements are sufficiently small so that no distinction is needed between the coordinates of a particle before and after deformation and that the displacement gradients are sufficiently small so that their products may be neglected. The last one implies that neither the temperature changes nor the stresses are too large. (In a practical case of line forming, this is not the case.)

Stress-Strain Relation

We think about thermal stresses due to a nonuniform temperature distribution. If we consider that a plate consists of small elements of material, the body must remain continuous, each element must restrain the distortion of its neighbors, in other words, stresses must rise. The total strains at each point consists of two components. The first one is uniform expansion proportional to the temperature rise $T = \theta - \theta_0$. If the coefficient of linear thermal expansion is denoted by α, this normal strain in any direction is equal to αT. The second component is the strains required to maintain the continuity of the body as well as those arising because of external load.

These strains are related to the stress by means of the Hook's law of linear isothermal elasticity. The total strains are the sum of the two components and are related as follow to the stress and temperature in any orthogonal coordinate system,x, y, z:

$$\varepsilon_{xx} = \frac{1}{E}[\sigma_{xx} - \nu(\sigma_{yy} + \sigma_{zz})] + \alpha T, \tag{3.18}$$

$$\varepsilon_{yy} = \frac{1}{E}[\sigma_{yy} - \nu(\sigma_{zz} + \sigma_{xx})] + \alpha T, \tag{3.19}$$

$$\varepsilon_{xx} = \frac{1}{E}[\sigma_{zz} - \nu(\sigma_{xx} + \sigma_{yy})] + \alpha T, \tag{3.20}$$

$$\varepsilon_{xy} = \frac{1}{2G}\sigma_{xy}; \quad \varepsilon_{yz} = \frac{1}{2G}\sigma_{yz}; \quad \varepsilon_{zx} = \frac{1}{2G}\sigma_{zx}. \tag{3.21}$$

The shear modulus G is related to Young's modulus E and Poisson's ratio ν by the equation,

$$G = \frac{E}{2(1+\nu)}. \tag{3.22}$$

The relation between the dilutation e and the sum of the normal stresses Θ is obtained from Equation 3.18 to Equation 3.20,

$$e = \frac{\Theta}{3k} + 3\alpha T, \tag{3.23}$$

where the bulk modulus k is

$$k = \frac{E}{3(1-2\nu)}, \tag{3.24}$$

and where

$$e = \varepsilon_{xx} + \varepsilon_{yy} + \varepsilon_{zz}, \tag{3.25}$$

$$\Theta = \sigma_{xx} + \sigma_{yy} + \sigma_{zz}. \tag{3.26}$$

The stresses explicitly can be expressed in terms of strains.

$$\sigma_{xx} = \lambda e + 2\mu\varepsilon_{xx} - (3\lambda + 2\mu)\alpha T, \tag{3.27}$$

$$\sigma_{yy} = \lambda e + 2\mu\varepsilon_{yy} - (3\lambda + 2\mu)\alpha T, \tag{3.28}$$

$$\sigma_{zz} = \lambda e + 2\mu\varepsilon_{zz} - (3\lambda + 2\mu)\alpha T, \tag{3.29}$$

$$\sigma_{xy} = 2\mu\varepsilon_{xy}; \quad \sigma_{yz} = 2\mu\varepsilon_{zx}. \tag{3.30}$$

The Lamé elastic constants λ and μ are related to E and ν as follows:

$$\lambda = \frac{\nu E}{(1+\nu)(1-2\nu)}; \quad \mu = \frac{E}{2(1+\nu)} = G. \tag{3.31}$$

It is necessary to enforce the requirement of mechanics and of geometry. The laws of mechanics are introduced by the equations of equilibrium; geometric consistency is stipulated through the strain-displacement relations.

Equations of Equilibrium

The following equations of equilibrium are based of purely mechanical considerations. In rectangular cartesian coordinates x,y,z:

$$\frac{\partial \sigma_{xx}}{\partial x} + \frac{\partial \sigma_{xy}}{\partial y} + \frac{\partial \sigma_{xz}}{\partial z} + X = 0, \tag{3.32}$$

$$\frac{\partial \sigma_{xy}}{\partial x} + \frac{\partial \sigma_{yy}}{\partial y} + \frac{\partial \sigma_{yz}}{\partial z} + Y = 0, \tag{3.33}$$

$$\frac{\partial \sigma_{xz}}{\partial x} + \frac{\partial \sigma_{yz}}{\partial y} + \frac{\partial \sigma_{zz}}{\partial z} + Z = 0. \tag{3.34}$$

where X, Y, and Z are the components of the body forces in the x, y, and z directions respectively.

The complementary components of the shear stress are equal,

$$\sigma_{xy} = \sigma_{yx}; \quad \sigma_{yz} = \sigma_{zy}; \quad \sigma_{zx} = \sigma_{xz}. \tag{3.35}$$

Strain-Displacement Relation

The strains are related to the displacements in the same manner as in isothermal elasticity since purely geometric consideration are involved;

$$\varepsilon_{xx} = \frac{\partial u}{\partial x}; \quad \varepsilon_{yy} = \frac{\partial v}{\partial y}; \quad \varepsilon_{zz} = \frac{\partial w}{\partial z}, \tag{3.36}$$

$$\varepsilon_{xy} = \frac{1}{2}\gamma_{xy} = \frac{1}{2}(\frac{\partial u}{\partial y} + \frac{\partial v}{\partial x}); \tag{3.37}$$

$$\varepsilon_{yz} = \frac{1}{2}\gamma_{yz} = \frac{1}{2}(\frac{\partial v}{\partial z} + \frac{\partial w}{\partial y}); \tag{3.38}$$

$$\varepsilon_{zx} = \frac{1}{2}\gamma_{zx} = \frac{1}{2}(\frac{\partial w}{\partial x} + \frac{\partial u}{\partial z}); \tag{3.39}$$

where u, v, and w are the components of the displacement vector in the x, y, and z directions respectively.

Solution of the Equations in Thermoelasticity

The problem of thermoelasticity consists of the deterministics of the following fifteen functions, with the temperature distribution obtained at Section 3.2.1 being assumed known and with appropriate boundary conditions:

6 stress components: σ_{xx}, σ_{yy}, σ_{zz}, σ_{xy}, σ_{yz}, σ_{zx};

6 strain components: ε_{xx}, ε_{yy}, ε_{zz}, ε_{xy}, ε_{yz}, ε_{zx};

3 displacement components: u, v, w.

However, within elasticity, deformation cannot be generated after the process. We need to think of plasticity to obtain deformation due to a moving heat source.

Plastic Problem

The following mathematical description on plastic problems is obtained from the book written by Masubuchi [25]. In order to reduce the space, Einstein's convention is used in the section. In other words, we implicitly sum over repeated indices.

Generally speaking, the rate of strain is described as follows:

$$\dot{\varepsilon}_{ij} = \dot{\varepsilon}_{ij}^{(e)} + \dot{\varepsilon}_{ij}^{(p)}, \tag{3.40}$$

where, superscripts (e) and (p) refer to the elastic and plastic strains, respectively.

The rate of plastic strain is assumed to be

$$\dot{\varepsilon}_{ij}^{(e)} = \Lambda \frac{\partial f}{\partial \sigma_{ij}}, \tag{3.41}$$

where Λ, f, and σ_{ij} are the proportional constant, the yield function, and the stress component, respectively.

The yield function is assumed to be

$$f(\sigma_{ij}, \zeta, \theta) = 0, \tag{3.42}$$

$$\zeta = \int \dot{\zeta}(\dot{\varepsilon}_{ij})dt, \tag{3.43}$$

where ζ is the parameter related to isotropic strain hardening of the material (no Bauschinger effect) and θ is the temperature.

After differentiating Equation 3.42 and using the result with Equation 3.41, the following relationship is obtained:

$$\dot{\varepsilon}_{ij}^{(p)} = \hat{G}|\frac{\partial f}{\partial \sigma_{ij}}\frac{\partial f}{\partial \sigma_{kl}}\dot{\sigma}_{kl} + \frac{\partial f}{\partial \sigma_{ij}}\frac{\partial f}{\partial \theta}\dot{\theta}|, \tag{3.44}$$

where,

$$\hat{G} = -\frac{1}{(\frac{\partial f}{\partial \varepsilon_{ij}^{(p)}} + \frac{\partial f}{\partial \zeta}\frac{\partial \zeta}{\partial \varepsilon_{ij}^{(p)}})\frac{\partial f}{\partial \sigma_{ij}}}. \tag{3.45}$$

von Mises's yield criterion is adopted. The temperature dependency can be expressed as follows:

$$f = \bar{\sigma} - c(\zeta, \theta), \tag{3.46}$$

where c is the parameter related to the strain hardening of the material and $\bar{\sigma}$ is defined as

$$\bar{\sigma} = \sqrt{\frac{3}{2}}\sqrt{\sigma'_{ij}\sigma'_{ij}}, \tag{3.47}$$

where σ'_{ij} is the stress deviation.

Using Equation 3.46, Equation 3.45 becomes

$$\hat{G} = -\frac{1}{\frac{\partial f}{\partial \varepsilon_{ij}^{(p)}}\frac{\partial f}{\partial \sigma_{ij}}}. \tag{3.48}$$

Here, introducing the rate of equivalent strain,

$$\dot{\bar{\varepsilon}}^{(p)} = \sqrt{\frac{2}{3}}\sqrt{\dot{\varepsilon}_{ij}^{(p)}\dot{\varepsilon}_{ij}^{(p)}}, \tag{3.49}$$

and then, using Equations 3.41 and 3.46, we have

$$1 = \frac{\partial f}{\partial \sigma_{ij}}\frac{\partial \bar{\varepsilon}^{(p)}}{\partial \varepsilon_{ij}^{(p)}}. \tag{3.50}$$

Hence, Equation 3.48 becomes

$$\hat{G} = -\frac{1}{\frac{\partial f}{\partial \varepsilon_{ij}^{(p)}} \frac{\partial \varepsilon_{ij}^{(p)}}{\partial \varepsilon^{(p)}}} = \frac{1}{H'}, \tag{3.51}$$

where,

$$H' = -\frac{\partial f}{\partial \bar{\varepsilon}^{(p)}}. \tag{3.52}$$

Finally, using Equations 3.46,3.47, and 3.51, the rate of plastic strain becomes

$$\dot{\varepsilon}_{ij}^{(p)} = \frac{1}{H'} \frac{3\sigma'_{ij}}{2\bar{\sigma}} (\frac{3\sigma'_{kl}}{2\bar{\sigma}} \dot{\sigma}_{kl} + \frac{\partial f}{\partial \theta} \dot{\theta}). \tag{3.53}$$

For the elastic part, we have:

$$\dot{\varepsilon}_{ij}^{(p)} = \frac{1-2\nu}{E} \dot{\sigma} \delta_{ij} + \frac{\dot{\sigma'_{ij}}}{2G} - \frac{1-2\nu}{E^2} \dot{E} \sigma \delta_{ij} - \frac{1}{2G^2} \dot{G} \sigma'_{ij} + \dot{\varepsilon}^{\theta} \delta_{ij}, \tag{3.54}$$

where, σ is the average hydrostatic stress, δ_{ij} is Kronecker's delta, ν is Poisson's ratio, E is Young's modulus, G is shear modulus, and ε^{θ} is the thermal strain caused by the temperature distribution.

The inverse relation of Equation 3.54 becomes:

$$\dot{\sigma}_{ij} = \frac{E}{1-2\nu} \dot{\varepsilon} \delta_{ij} + 2G \dot{\varepsilon}_{ij} - \frac{3G\sigma'_{ij}\sigma'_{kl}\dot{\varepsilon}_{kl}}{\bar{\sigma}^2 (H'/3G+1)} + \dot{\sigma}_{ij}^{\theta}, \tag{3.55}$$

where,

$$\varepsilon = \frac{1}{3} \varepsilon_{ij}; \tag{3.56}$$

$$\dot{\sigma}_{ij}^{\theta} = -\frac{E}{1-2\nu} \dot{\varepsilon}^{\theta} \delta_{ij} + \frac{\sigma}{E} \dot{E} \delta_{ij} + \frac{\sigma'_{ij}}{G} [1 - \frac{1}{(H'/3G+1)}] \dot{G} - \frac{3G\sigma'_{ij}}{\bar{\sigma} H'} \frac{\partial f}{\partial \theta} [1 - \frac{1}{(H'/3G+1)} \dot{\theta}]. \tag{3.57}$$

In this equation, σ_{ij}^{θ} consists of the terms related to the rate of thermal strain and the temperature dependency of the material properties and of the yield criterion.

If an element in question is plastic, the element takes one of the following states

during the next increment:

$$f = 0 \quad \dot{f}' < 0 \quad (unloading), \tag{3.58}$$

$$f = 0 \quad \dot{f}' = 0 \quad (neutral), \tag{3.59}$$

$$f = 0 \quad \dot{f}' > 0 \quad (loading), \tag{3.60}$$

where,

$$\dot{f}' = \frac{\partial f}{\partial \sigma_{ij}} \dot{\sigma}_{ij} + \frac{\partial f}{\partial \theta} \dot{\theta}. \tag{3.61}$$

If the yield function does not move and does not change the size, the loading in the element does not occur.

3.3 Finite Element Formulation

The above mentioned constitutive equations of the thermo-elastic-plastic problem are too complicated to solve them analytically for arbitrary boundary conditions. The FEM analysis is a strong tool to solve them numerically. In FEM analysis, we need to solve two problems independently: thermal and stress-strain analyses. The formulations of the problem are described as follows.

3.3.1 Thermal Analysis

A finite element formulation for uncoupled problem of heat transfer was developed by Zienkiewicz and Parekh [55]. They applied the Galerkin weighted residual method to the governing differential equation of heat transfer to obtain the finite element equations which can be represented in the following matrix form:

$$[C_T]\{\dot{T}\} + [K_T]\{T\} = \{Q\}, \tag{3.62}$$

where, $[C_T]$ is the heat capacity matrix, $[K_T]$ is the heat conduction matrix, $\{T\}$ is the column vector of the nodal point temperatures, $\{\dot{T}\}$ is the time derivative of

temperature, Q is the column vector of the nodal point heat sources.

The governing incremental isoparametric finite element equation for heat conduction analysis has been derived by Bathe [2] and is:

$$({}^{t}K^{k} + {}^{t}K^{c} + {}^{t}K^{r})\Delta\theta^{i} = {}^{t+\alpha\Delta t}Q + {}^{t+\alpha\Delta t}Q^{c(i-1)} + {}^{t+\alpha\Delta t}Q^{r(i-1)} - {}^{t+\alpha\Delta t}Q^{k(i-1)}, \quad (3.63)$$

or

$$ {}^{t}\bar{K}\Delta\theta^{(i)} = {}^{t+\alpha\Delta t}\bar{Q}^{(i-1)}, \quad (3.64)$$

where ${}^{t}\bar{K}$ is the effective conductivity matrix at time t consisting of the conductivity, nonlinear convection, and radiation matrices; ${}^{t+\alpha\Delta t}Q$ is the heat flow vector including the effect of surface heat flow inputs, internal heat generation, and temperature dependent heat capacity; ${}^{t+\alpha\Delta t}\bar{Q}^{i-1}$ is the effective heat flow vector; and $\Delta\theta^{(i)}$ is the increment in the nodal point temperature in iteration i,

$$ {}^{t+\alpha\Delta t}\theta^{(i)} = {}^{t+\alpha\Delta t}\theta^{(i-1)} + \Delta\theta^{(i)}. \quad (3.65)$$

Equation 3.63 is the general incremental heat flow equilibrium equation that is valid for linear and nonlinear analysis. The value of α which is $0 \leq \alpha \leq 1$ was chosen to obtain optimum stability and accuracy in the solution.

Convection and radiation boundary conditions are taken into account by including the matrices ${}^{t}K^{c}$ and ${}^{t}K^{r}$ and the vectors ${}^{t+\alpha\Delta t}Q^{c}$ and ${}^{t+\alpha\Delta t}Q^{r}$ in the heat flow equilibrium equations. Additional external heat flow input on the boundary is specified in ${}^{t+\alpha\Delta t}Q$ as surface heat input.

In transient analysis, a numerical time integration scheme must be employed. A family of one step methods can be used, in which

$$ {}^{t+\alpha\Delta t}\dot{\theta} = \frac{({}^{t+\Delta t}\theta - {}^{t}\theta)}{\Delta t}, \quad (3.66)$$

$$ {}^{t+\alpha\Delta t}\theta = (1-\alpha){}^{t}\theta + \alpha{}^{t+\Delta t}\theta, \quad (3.67)$$

where $0 \leq \alpha \leq 1$.

This scheme was found to be unconditionally stable for $\alpha \geq \frac{1}{2}$ and to generally give better solution when $\alpha = 1$ (Euler Backward method). The modified Newton iteration is guaranteed to converge provided the time step Δt is small enough.

Utilizing an FEM program eliminates the assumptions required to solve the heat flow problem mathematically in Section 3.2.1. For example, the size of the heat source is not necessarily small in the thermal analysis by FEM. Moreover, the heat flux distribution of the heat source can be considered.

However, since the temperature distribution generated by a moving heat source is usually very steep, the time increment and the spatial mesh size must be small enough to make the calculation converge accurately. This characteristic of the calculation by FEM makes the thermal analysis of a moving heat source time consuming and memory demanding.

3.3.2 Stress-Strain Analysis

The model was developed by Snyder and Bathe [41] and is described as follows.

The governing incremental finite element equation for the problem can be written as:

$$^{t+\Delta t}K^{(i-1)}\Delta U^{(i)} = {}^{t+\Delta t}R - {}^{t+\Delta t}F^{(i-1)}, \tag{3.68}$$

where $^{t+\Delta t}K^{(i-1)}$ is the tangent stiffness matrix at time $t+\Delta t$ which includes the linear and nonlinear strain stiffness matrices; $^{t+\Delta t}$R is the vector of externally applied force at time $t + \Delta t$; $^{t+\Delta t}F^{(i-1)}$ is the vector of nodal point force due to element internal stress at time t; and $\Delta U^{(i)}$ is increment in nodal point displacement in iteration i.

The term $^{t+\Delta t}F^{(i-1)}$ can be evaluated for the materially nonlinear as follows:

$$^{t+\Delta t}F^{(i-1)} = \int_V {}^{t}B_L^{T\,t+\Delta t}\sigma^{(i-1)}dV, \tag{3.69}$$

where $^{t}B_L^T$ is the constraint strain-displacement transformation matrix; and $^{t+\Delta t}\sigma^{(i-1)}$ is the stress vector corresponding to the nodal point displacement $^{t+\Delta t}U^{(i-1)}$.

A basic assumption in the formulation of the model and of plasticity is that the

infinitesimal total tensor for increments can be expressed as the sum of elastic, plastic, and thermal strain increments:

$$ {}^t\dot{e}_{ij} = {}^t\,\dot{e}^E_{ij} + {}^t\,\dot{e}^P_{ij} + {}^t\,\dot{e}^{TH}_{ij}, \tag{3.70} $$

where ${}^te_{ij}$ is the component of infinitesimal total strain tensor at time t; ${}^te^E_{ij}$ is the component of infinitesimal elastic strain tensor at time t; ${}^te^P_{ij}$ is the component of infinitesimal plastic strain tensor at time t; and ${}^te^TH_{ij}$ is the component of infinitesimal thermal strain tensor at time t.

This assumption allows the distinction between time dependence and time independence of inelastic strains. At any time t, the stress is given by the constitutive law for isotropic thermal elastic material for small displacement analysis as follows:

$$ {}^t\sigma_{ij} = {}^t\,C^E_{ijrs}({}^te_{rs} - {}^t\,e^P_{rs} - {}^t\,e^{TH}_{rs}), \tag{3.71} $$

where ${}^tC^E_{ijrs}$ is the component of elastic constitutive tensor at time t.

The thermal strains are:

$$ {}^te^{TH}_{rs} = {}^t\,\alpha_m({}^t\theta - \theta_R)\delta_{rs}, \tag{3.72} $$

where ${}^t\alpha_m$ is the mean coefficient of thermal expansion; ${}^t\theta$ is material temperature at time t; θ_R is the reference temperature at which thermal strains are zero; and δ_{rs} is Kronecker delta.

The general form of the yield or loading function for multiaxial stress condition in non-isothermal condition is;

$$ {}^tF = {}^t\,F({}^t\sigma_{ij}, {}^t\,\alpha_{ij}, {}^t\,\sigma_y), \tag{3.73} $$

where ${}^t\alpha_{ij}$ and ${}^t\sigma_y$ depend on the history of plastic deformation and temperature. When stress and strain axes are coincident, the infinitesimal plastic strain increment

vector is normal to the current yield surface. The result can be expressed as:

$$e_{ij}^{P} =^{t} \lambda \frac{\partial^{t} F}{\partial^{t} \sigma_{ij}}, \tag{3.74}$$

where,

$$e_{ij}^{P} =^{t+\Delta t} e_{ij}^{P} -^{t} e_{ij}^{P}, \tag{3.75}$$

and ${}^{t}\lambda$ is positive scalar variable.

The von Mises yield function for non-isothermal kinematic hardening can be written as:

$$^{t}F = \frac{1}{2}(^{t}s_{ij} -^{t} \alpha_{ij})(^{t}s_{ij} -^{t} \alpha_{ij}) - \frac{1}{3}^{t} \sigma_{y}^{2}, \tag{3.76}$$

where ${}^{t}\sigma_{y}$ is yield stress; and ${}^{t}\alpha_{ij}$ is tensor measuring translation of the yield surface.

A governing incremental stress-strain relation at time $t + \Delta t$ and iteration i can be written as:

$$^{t+\Delta t}\sigma^{(i)} =^{t+\Delta t} C^{E}[^{t+\Delta t}e^{(i)} -^{t+\Delta t} e^{P(i)} -^{t+\Delta t} e^{TH(i)}], \tag{3.77}$$

where,

$$^{t+\Delta t}e^{P(i)} =^{t} e^{P} + \int_{t}^{t+\Delta t(i)} d^{t+\Delta t}e^{P}. \tag{3.78}$$

3.4 Inverse Problem to Obtain the Heat Process Conditions

As shown in the above, the description of mechanism of metal forming process due to a moving heat source is complicated. Therefore, it is difficult to obtain the heating processes to get a desired shape.

In the thermal analysis, we obtain the temperature distribution as a function of three coordinates of position and time. Given the temperature history, solving the thermo-elastic-plastic problem, we can determine the deformation due to a moving heat source for arbitrary heating conditions. As described above, the mathematical

description is too complicated to obtain the function representing the relationship between the heat condition of a moving heat source and the deformation.

Certainly, we can obtain the deformation due to a moving heat source with a thermo-elastic-plastic FEM program. The problem of this procedure is that it requires much computation time. The causes of the problem are as follows:

1. Since the temperature distribution due to a moving heat source is very steep, the spatial mesh size along the direction of the heat source's movement must be fine enough for FEM calculation to converge in both the heat transfer and stress-strain analyses.

2. Because of the same reason mentioned above, the time step must be small enough in both the heat transfer and stress-strain analyses.

3. We need to calculate the heat transfer analysis first, followed by the stress-strain analysis.

Even though it is possible to obtain the resultant deformations for arbitrary heating conditions with FEM analysis, we cannot solve the relationship inversely.

Therefore, we need to a new algorithm to obtain a set of heat process conditions for a desired shape of a plate.

3.5 Summary

In this chapter, I have presented the mathematical description of thermal analysis, thermo-elastic analysis, plastic analysis, and their implementation by FEM. In order to obtain the heating conditions for a given deformation, we need to solve the problem inversely. If we can approximate the relationship between the heating conditions and the deformation mathematically, we can invert the function to obtain the heating conditions for a given deformation. Although FEM is good at solving thermo-elastic-plastic analysis, it can merely obtain a deformation for a given heating condition. There is no way that FEM analysis can solve the problem inversely. Therefore, we

need to invent an algorithm to design the heating process parameters to obtain a desired shape.

Chapter 4

Conceptual Design Procedure of Process Parameters

In this chapter, the newly developed heating condition algorithm is presented. As pointed out in the previous chapter, a mathematical analysis has not been found for such complex inverse problems and the FEM is far too tedious for a tries-and-error or steepest decent method.

4.1 Controllable Heat Process Parameters

The heat process parameters which we can control are the positions or areas of heating lines, the speed of a moving heat source, the heat input intensity and its distribution, and the heat spot diameter. On the other hand, we have several material properties which are constants for a given material: yield stress, heat conductivity, and one which is difficult to control: heat absorption rate. Choosing one kind of heat source, for example, one kind of laser beam and one material fixes several parameters, such as heat input intensity and its distribution, heat spot diameter, yield stress, heat conductivity, and heat absorption rate. Then, only the positions of heating lines and the process speed are process variables which can be controlled.

This situation is similar to the manual line heating process used in shipyards. An operator controls the traverse speed of a gas flame, decides where to put heating lines,

and when and where cool down the plate with water. However, enforced cooling has not been dealt with in my work, since a proposed automated laser forming machine does not have such functions.

The controllable parameters are the following:

1. Heating Line Positions

2. Heat Input Intensities for the Lines

The positions of heating lines are critical for the forming process. Line heating operators for many years have had to learn where they should put heating lines. On the other hand, it is difficult to determine the heating line positions with computers or conventional algorithms. Generally speaking, the heating pattern can be any two dimensional shape on a plate such as a line, a circle, and a curve. However, if we consider of the general case of the heat pattern, the problem would be too complicated. Then, a line was chosen as the heating pattern to simplify the problem.

The heat input intensity is determined by a several dependent parameters: process speed, the intensity of a heat source, the area of heat input (spot size for a laser beam). In fact, some researchers have investigated the relationship between the output of deformation and some heat input intensity parameters. Masubuchi et al. [27] has found a good relationship between the angular distortion and the empirical parameter of $P/(t\sqrt{v})$ in laser forming of steel plates by a CO_2 laser beam , where P is the power of a laser beam, t is thickness of a plate, and v is the velocity of a process. The result is shown in Figure 4-1. This kind of information is necessary for designing heat process conditions as a database.

For manual line heating, operators adjust the process speed only, since they cannot change the intensity of a gas flame as they move the torch. Even in an automated line heating with a laser beam, it is very unusual to adjust the output power of the laser beam during a heat process.

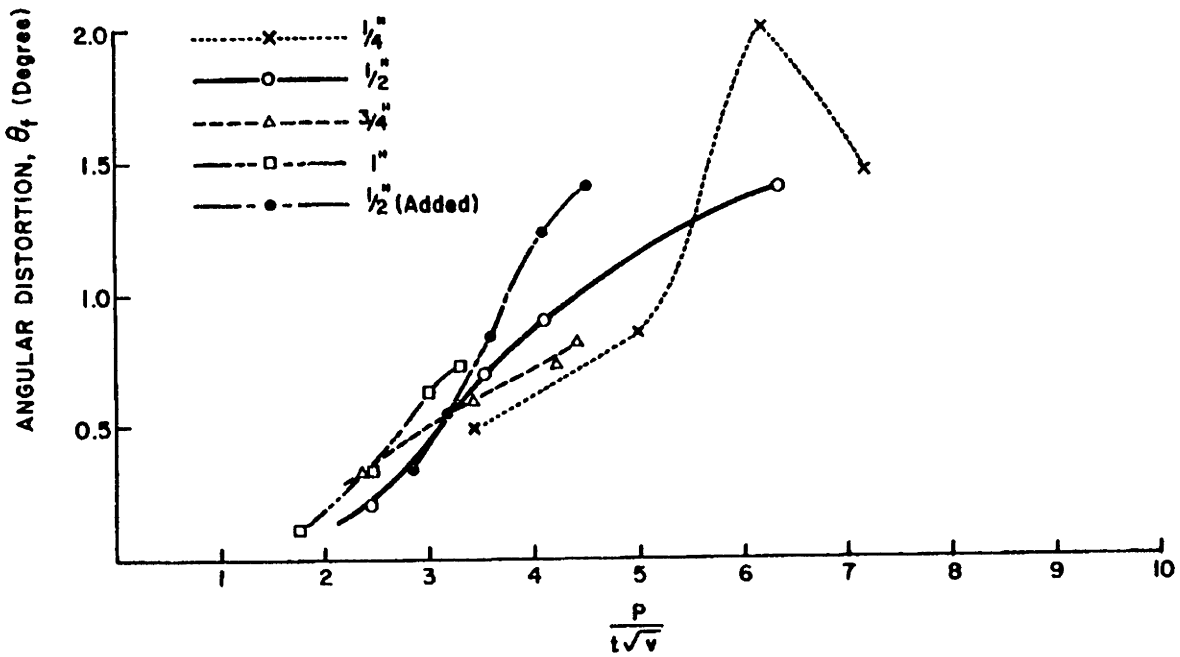


Figure 4-1: Relation between angular distortion and the parameter, $P/(t\sqrt{v})$ obtained by Masubuchi et al. [27]

4.2 Function of the Process Design Program

As mentioned in the previous section, the output deformation due to a moving heat source is the function of the positions of heating lines and the intensities of heating lines. Ideally speaking, a process design program needs to determine both the positions and the intensities of heating lines. However, after the review of literature survey, I decided to give up the determination of the positions, realizing that the existing computation algorithm cannot deal with the determination of the positions.

The program can decide the magnitudes of heat intensities for heating lines, when it is given the target shape from designers of a product and the positions of heating lines as shown in Figure 4-2. Even though the positions of heating lines have to be determined prior to computation, the program would be still helpful for the automation of the forming process by a moving heat source. After the program finishes determining the heat intensity for each heating line, the results will be easily transformed into the information for the movement and the process parameters of a process cell

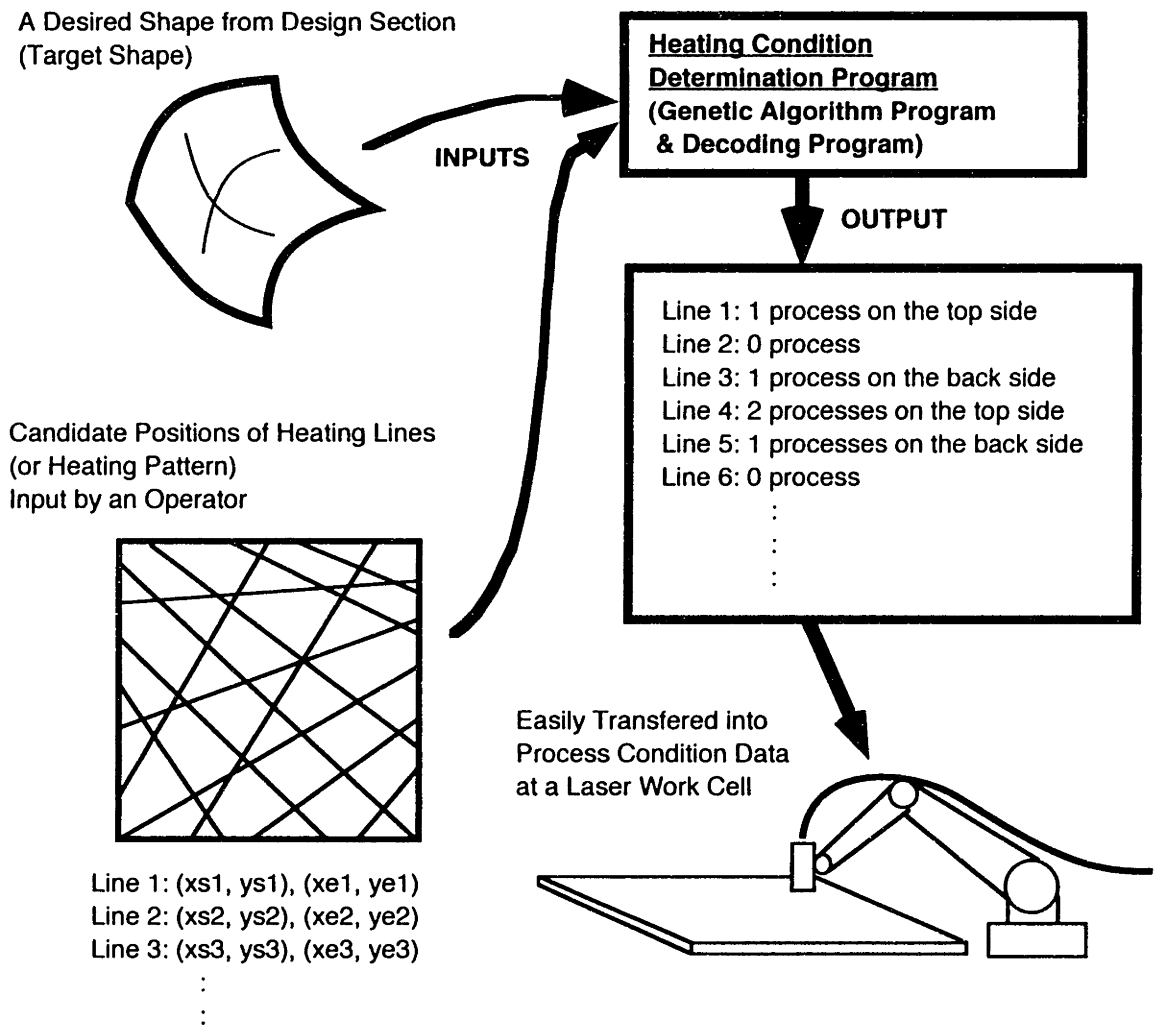


Figure 4-2: Function of a process condition design program

which consists of, for example, a robot and a laser beam oscillator.

The determination of the heating line positions requires a learning process. An operator who uses the program has to determine the heating line positions based on knowledge or experience. Therefore, the operator of the program must learn to guess the heating pattern to obtain a desired shape.

After the operator decides upon the positions, the program can optimize the heating intensities for all heating lines. What the program does is to find the best solution for a given set of heating lines. Therefore, if the guess by the operator is good, the program gives a set of heat intensities with which to form a plate into a desired shape with good accuracy. If the guess is not good, the program produces a

set of heat intensities with less accuracy. In other words, the accuracy of the result would depend strongly on the accuracy of the guess made by the operator.

4.3 Comparison among Optimization Methods

Determinating the heat intensities for heating lines is an optimization processes. The optimization process uses the objective function to be maximized. In this case, the objective function is related to the summation of errors between the displacement of a desired shape and that of an expected shape. Since the objective function usually must increase as the parameters approach to the optimized values, the inverse of the sum of errors can be used as an objective function. The comparison is based on the book written by Goldberg [12].

4.3.1 Conventional Optimization Methods

Conventional optimization methods are classified into two categories: calculus-based method and numerical scheme. In this section these two optimization methods are examined in terms of suitability for the heating condition program.

Calculus-Based Methods

Calculus-based methods have been studied extensively. These subdivide into two main classes: indirect and direct. Indirect methods seek local extrema by solving the usually nonlinear set of equations resulting from setting the gradient of the objective function equal to zero. This is the multidimensional generalization of the elementary calculus notion of extremal points. Given a smooth, unconstrained function, finding a possible peak starts by searching for those points with slopes of zero in all directions.

On the other hand, direct search methods seek local optima by hopping on the function and moving in a direction related to the local gradient. This is simply the notion of hill climbing: to find the local best, climb the function in the steepest permissible direction.

First, both methods are local in scope; the optima they seek are the best in a neighborhood of the current point. Also, once the lower peak is reached, further improvement must be sought through random restart or other trickery. Second, calculus-based methods depend upon the existence of derivatives. Even if we allow numerical approximation of derivatives, this is a severe shortcoming.

Numerical Schemes

Many numerical schemes have been considered. The idea is fairly straightforward; within a finite search space, or a discretized infinite search space, the search algorithm starts looking at objective function values at every point in the space, one at a time. Although the simplicity of this type of algorithm is attractive, and enumeration is a very human kind of search when the number of possibilities is small, such schemes must ultimately be discounted for one simple reason: lack of efficiency.

4.3.2 Genetic Algorithm

Genetic algorithms (GAs) are search algorithms based on the natural mechanics of natural genetics and selection [12]. They combine survival of the fittest among string structures with a structured yet randomized information exchange to form a search algorithm with some of the innovative flair of human search.

Genetic algorithm is most suitable for constructing the design method for a forming process by a moving heat source, because obtaining the derivatives of the objective function that is necessary for calculus-based optimizing methods is difficult.

4.4 Heat Process Design Algorithm by GA

If a simple plate model is used to express the relationship between displacement and stresses or strains such as the thin plate theory, the distribution of strains or stresses can be determined analytically not if plastic. When this is the case, conventional non-linear optimization methods can be used for determining the heating process conditions for a desired shape.

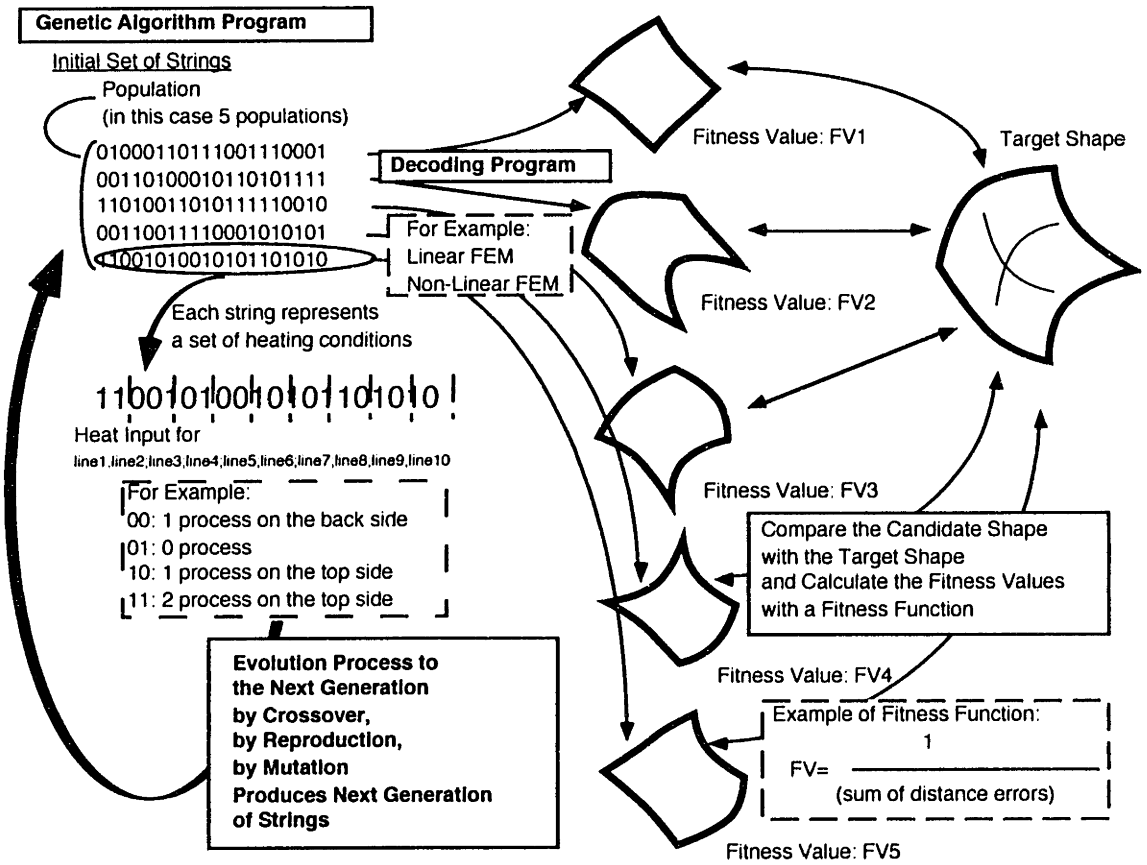


Figure 4-3: Procedure of heating condition determination by Genetic Algorithm

However, there is no guarantee to get an accuracy good enough to enable us to construct the laser forming machine. A complicated model that prevents the stress or strain distribution being obtained analytically for a pre-determined shape seems to be needed for the prediction of the deformation.

Moreover, even if the stress or strain distribution is obtained analytically, translating the obtained stress or strain distribution to proper heating conditions is an inverse problem and is difficult to solve.

The design procedure for a heating condition with GA is shown in Figure 4-3. The process consists of two parts: an optimizing program and a decoding program. The following sections explain the functions of the parts.

4.4.1 Optimizing Part

The optimizing program is the engine of the optimizing procedure. It is going to determine the heat input for each given heating line in order to form a desired shape.

Encoding Scheme

When the GA is utilized for an optimizing procedure, the heat conditions must be coded into the string of the combinations of "0" and "1." One string consists of sets of bits that are "0" or "1." Each set of bits express a pattern of the heat intensity. The positions of the sets of bits correspond to the positions of the heating lines given by the operator.

For example, as shown in Figure 4-3, each set of two bits divided by a dotted line expresses the heat procedure for a heating line as follows:

- 00: one heating process on the bottom surface,
- 01: no heating process,
- 10: one heating process on the top surface,
- 11: two heating processes on the top surface.

If we have more options of heating processes, more bits can be used to express the heating condition for a heating line. If eight bits are used, 256 levels of heating conditions can be expressed.

On the other hand, the order of a set of bits from the beginning represents the position of heating line. The first two bits express the heating condition for "heating line 1" whose position is defined by the operator.

Fitness Value and Fitness Function

The comparison method between the expected shape and the target shape is called a fitness function. The simplest fitness function can be given the following equation:

$$FV = \frac{1}{1 + \sum_{i=1}^{N} |z_{ti} - z_{ei}|/(t \cdot N)} \tag{4.1}$$

where, FV is the fitness value, z_{ti} is the ith displacement value of the target shape, z_{ei} is a corresponding displacement value of the trial shape, N is the number of reference points, and t is thickness of the plate.

The value of FV increases when the error between the target shape and the expected shape approaches to zero.

Evolution

The evolution is achieved by the following four steps:

1. Initialization: A given population consists of n individuals. Each is characterized by its genotype consisting of i genes, which determine the vitality or fitness for survival. Each individual's genotype is represented by a binary bit string, representing the object parameter value by means of an encoding scheme.

2. Selection: Two parents are chosen with probabilities proportional to their relative position in the current population, each measured by its contribution to the mean objective function value of the generation.

3. Reproduction: Two different preliminary offsprings are produced by recombination of two parental genotypes by means of crossover at a given recombination probability. Only one of those offsprings is actually taken into further consideration. Selection step and reproduction step are repeated until n individuals represent the next generation.

4. Mutation: The offspring encounters further modification with a given fixed and small probability by means of point mutations working on individual bits by reversing a one to a zero or vice versa.

The average fitness value of the generation will increase on repeating the above mentioned evolutionary procedure. One of the individuals among a generation finally reaches to the global optimum after many generations of evolution.

4.4.2 Decoding Part

A decoding program must be utilized to compute a fitness value for a string. It drives a string to a physical shape. In other words, a string is transformed with a decoding program into an expected shape by a set of heating procedures expressed by the string. The decoding program is necessary to calculate a fitness value for each string. Once the expected shape is obtained, it is compared with the target shape.

Theoretically speaking, we can use a precise thermal-elastic-plastic FEM code to decode the strings into deformations. However, the decoding program with such a precise program is not practical because of the calculation time required to obtain a deformation. Since the optimizing part needs to use the decoding program so many times to approach the optimized solution, precision of the decoding program depends on speed of calculation of computer to be used for the calculation.

4.5 Summary

The newly developed heat determination algorithm was presented in this chapter. The set of heating conditions is encoded into a string. In the coding procedure, the heating conditions have to be discretized. The string is decoded into an expected shape and then the expected shape is compared to the target shape with the fitness function. The strings evolve by reproduction, crossover, and mutation according to the fitness value until the best string satisfies the required accuracy.

The accuracy of the algorithm strongly depends on the accuracy of a decoding program. If we do not care about calculation time, we can obtain the solution accurately enough for the procedure. However, as we have a practical time constraint for the calculation, we have to compromise the accuracy of the program.

Chapter 5

Implementation of a Process Determination Program

As shown in the previous chapter, in principle Genetic Algorithms (GAs) can solve and determine the heat input intensity for a given set of heating lines to obtain the final shape that best fits a target shape. Theoretically speaking, a thermo-elastic-plastic FEM program can be used to achieve an accurate prediction if some computer which can calculate the deformation is used within a reasonable short time. This means that as the calculation speed of the computer increases, the program can be updated for better accuracy.

In order to demonstrate the ability of the algorithm, I developed the program following the concept of the genetic heat determination procedure presented in the previous chapter. The source code is presented at Appendix A.

The simplest FEM model was used only because it can reduce calculation time and can be implemented easily. After investigating the usefulness of the algorithm for heat condition determination and the calculation time for the most simple material model, one can easily replace the subprogram with that based on a more precise material and mechanics model.

5.1 Flowchart of the Program

The flowchart of the program is shown in Figure 5-1. The optimization process is done by a GA program while the calculation of deformations is conducted by an FEM program in order to obtain the fitness values.

Figure 5-2 shows the organization of the program. The functions of the subroutines are explained in Appendix A.

The program consists of two parts: a decoding program and an optimizing program. A linear FEM program is used as a decoding program. The linear FEM program calculates the deformation for the given load conditions. GA is used as an optimizing program. The GA generates sets of heating condition magnitude for a given set of heating line positions and compares the resulting deformation of a plate with a predetermined shape. The input program transforms a set of heating line positions into load forces onto nodes based on the set of heating condition magnitude generated by the GA.

5.2 Optimizing Program with Genetic Algorithm

Several GA drivers can be obtained with their source codes via the Internet. D.L. Carroll's FORTRAN Genetic Algorithm Driver version 1.6.2 [5] developed by D. L. Carroll at the University of Illinois was chosen among them, simply because it is written in FORTRAN. This choice made it easy to combine the GA driver and the FEM subprogram since both are written in the same computer language.

This program is designed to be used as a GA driver for many applications. Users can apply it for their use with just changing an evaluation function subroutine. Every time the GA driver generates a string representing the heat input values for all predetermined heating lines, the subroutine calls the FEM subprogram to calculate the deformation for the given heating conditions. Then, the subroutine calculates the amount of the difference between the pre-determined shape and the shape obtained by FEM calculation. This value is also translated to fitness value in the subroutine

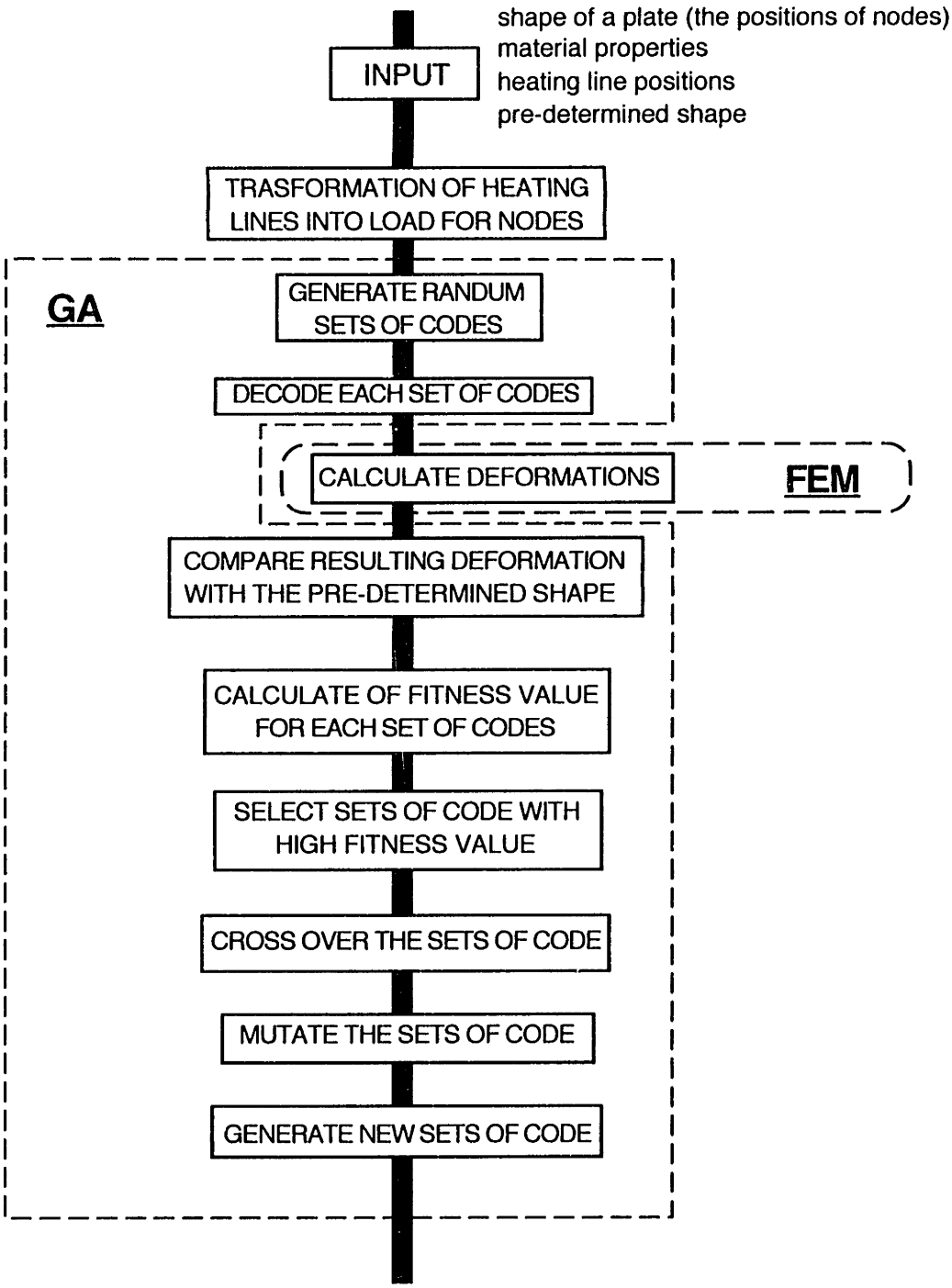


Figure 5-1: Flowchart of program

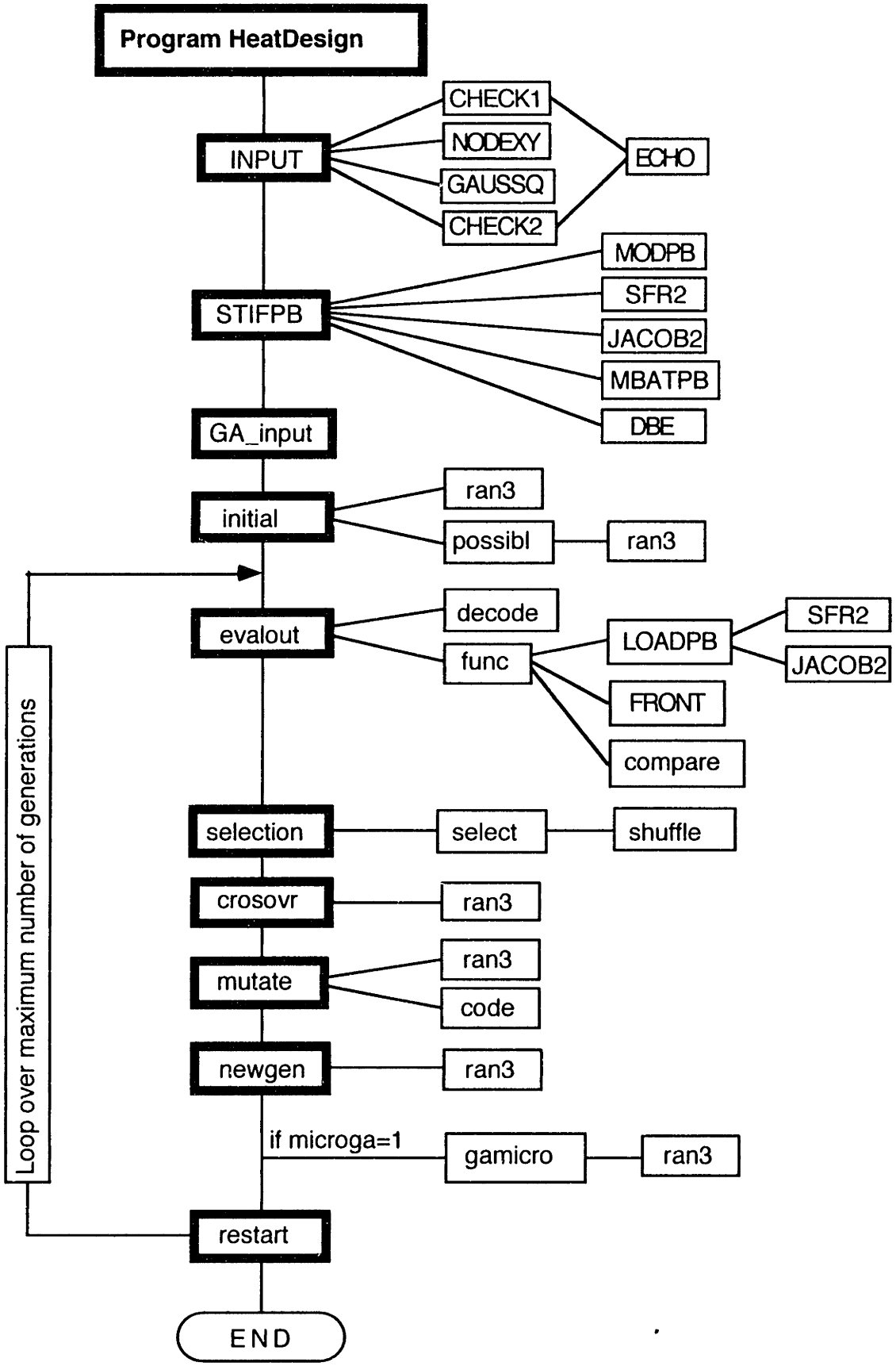


Figure 5-2: Program organization

for the use in the GA driver.

5.2.1 Representation of Heating Conditions with Coding

Heating conditions of laser line heating are considered as a set of parameters. Each parameter indicates the heat input pattern for a heating line. The order of parameters relates with the positions of heating lines on a plate. However, in a GA driver we need to express a set of heat input values with a coding of parameters called a string not the parameters themselves. The value of a parameter is expressed as a combination of 0 and 1. These procedures are called coding.

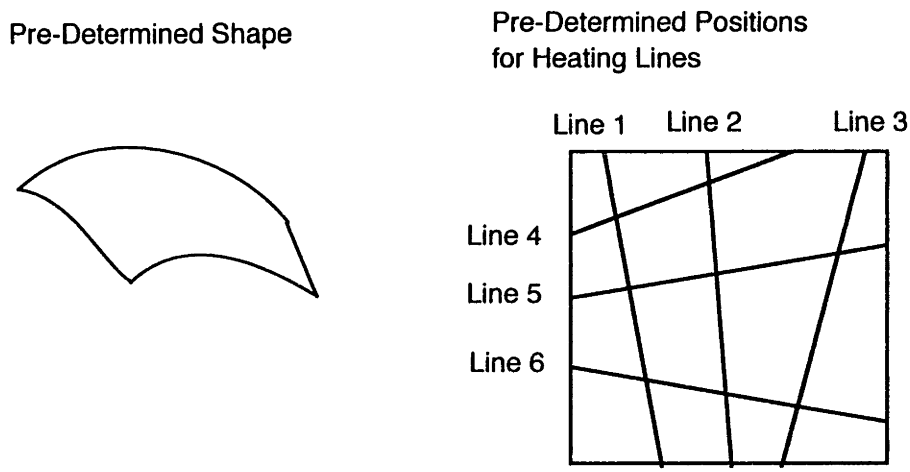
For example, suppose we have six heating lines on a plate and want to determine each heat input value for a heating line using a GA driver. (See Figure 5-3.) In the string of parameters the first parameter is the heat input value for heating line 1, the second is for heating line 2, and so on. The nth parameter is for heating line n. Then the string of parameters are coded into the string of bits. The range of the parameters is related to the numbers of digits required for coding. In the example shown in Figure 5-3, all parameters have the range of 4. Therefore, one parameter needs two digits to be expressed. After we get the string of bits, a GA driver for a optimization process can be used.

5.2.2 Fitness Function

Fitness values are used in a GA driver in order to select good strings from generation to generation. Considering the displacements calculated by FEM at the node points, the fitness value, FV, is calculated by the following equation:

$$FV = \frac{1}{1 + \sum_{i=1}^{N} |DT_i - DC_i|/(t \cdot N)} \tag{5.1}$$

where DT_i is the target displacement value for point i among N points, DC_i is the resulting value for point i by FEM analysis, N is the number of reference points, and t is the thickness of the plate.



(a) Input for the Calculation

Heating Lines	1	2	3	4	5	6
Heat Inputs	h1	h2	h3	h4	h5	h6
Coded bits of strings	01	00	10	11	01	11

(b) Structure of String of Heat Input for Heating Lines

Figure 5-3: Coding of heating conditions in the program

FV is large for small difference between the target shape and the calculated shape by the FEM subprogram.

5.2.3 Input of Heating Lines

In order to apply bending moments on proper nodes automatically based on the heating line position so that the program can be used as a practical heat condition determination tool, the following conditions have to be satisfied:

1. Heating lines can be applied at arbitrary positions on a plate including diagonal positions,

2. Bending moments have to be applied on a node as precalculated functions of magnitudes of heating processes in the FEM subprogram.

The first one is definitely required for practical uses of the program. The second condition is required since the program has to calculate the deformations fast. Otherwise, the FEM subprogram would need to calculate the bending moments for nodes each time the GA driver generates a set of heating processes. This would be a time-consuming process.

5.3 Linear FEM Program as a Decoding Program

In order to use a GA as an optimizing program, a set of heating process conditions must be encoded by characters of 0 and 1. A decoding program is necessary to compare an expected final shape from a set of heating process conditions to a target final shape. A thermal-elastic-plastic FEM program can be used as a decoding program for the maximum accuracy. However, since such a program required much computation time for obtaining one result, to use an thermal-elastic-plastic program is not practical. During the optimizing procedure, the decoding program must calculate thousands of results of expected final shapes.

Considering the performance of current computers, we need to use a simple program that requires less time to predict a final shape for a set of heating process

conditions. In order to reduce the calculation time, we need to make a compromise in terms of accuracy.

A linear FEM program was used as a decoding program to predict final shapes for sets of heating process conditions. Although a linear FEM program cannot directly simulate a deformation due to a moving heat source, it can predict a final shape if we transform residual plastic strains into loads.

The decoding program is based on the program presented in the book "Finite Element Programming" by E. Hinton and D. R. J. Owen, Academic Press Inc. (London) Ltd., 1977[15]. The program is written in FORTRAN. Its formulation is based on the so-called "thin plate" theory where out-of-plane shear deformations are neglected. In the formulation of the plate element, the author of the program use the assumptions adopted by Mindlin [30]: the deflections of the plate are small; normals to the midsurface before deformation remain straight but not necessarily normal to the midsurface after deformation; stresses normal to the midsurface are negligible irrespective of the loading. Under these assumptions, the decoding program is expected to be accurate only for small displacements.

5.3.1 Assumptions of the Decoding Program

To utilize a linear FEM program as a decoding program incurs two significant assumptions. These assumptions are the causes of error between the actual final shape of a plate and the expected final shape calculated by the linear FEM program.

Linearity of Addition of Plastic Strain Distributions

When considering the meaning of laser forming or line heating, we realize that this is just adding plastic bending strain distributions due to a moving heat source onto a plate. The added plastic strain distribution causes deformation. Although plastic strain is produced by complicated mechanism due to a moving heat source, we can add a plastic strain distribution caused by a heating procedure to a initial plastic strain because the linearity of addition of deformation holds as shown by Ueda et

al [45]. This assumption makes the linear FEM program ignore the effect of the process order onto a plate on the final shape.

Ignorance of Inplane Shrinkage as the First Order Approximation

As the first order approximation, I focused on bending moment caused by transverse shrinkage, since the bending moment has the direct effect on out-of-plane deformation. This assumption was also employed by Nomoto [37] for the construct of a line heating simulator. This assumption will have to pay a price of a decrease in accuracy for an decrease of calculation time. However, as I wanted to concentrate on the establishment of a design algorithm, I decided to pay the price.

5.3.2 Isoparametric Element and Geometric Data

Since the program uses isoparametric plate elements with eight nodes, it can deal with any kind of shape as shown in Figure 5-4 (a) and (b). Each node has three degrees of freedom: parallel displacement in z-axis, rotation in xz-plane, and rotation in yz-plane as shown also in Figure 5-4 (c).

5.3.3 Boundary Conditions

A plate is almost free during a practical forming process. Only the gravity force is applying onto the plate. On the other hand, we need to prevent the plate from moving in the linear FEM program. Three edge nodes of a plate are fixed in the displacement in z-axis in order to simulate a forming process by a moving hear source.

5.3.4 Load Conditions

Since the FEM decoding program uses isoparametric plate elements with eight nodes, the node of an element has three degrees of freedom: displacement in z-axis, rotation in xz-plane, and rotation in yz-plane. Then one force and two bending moments related to the three degrees of freedom can be applied as loads onto a node as shown in Figure 5-5.

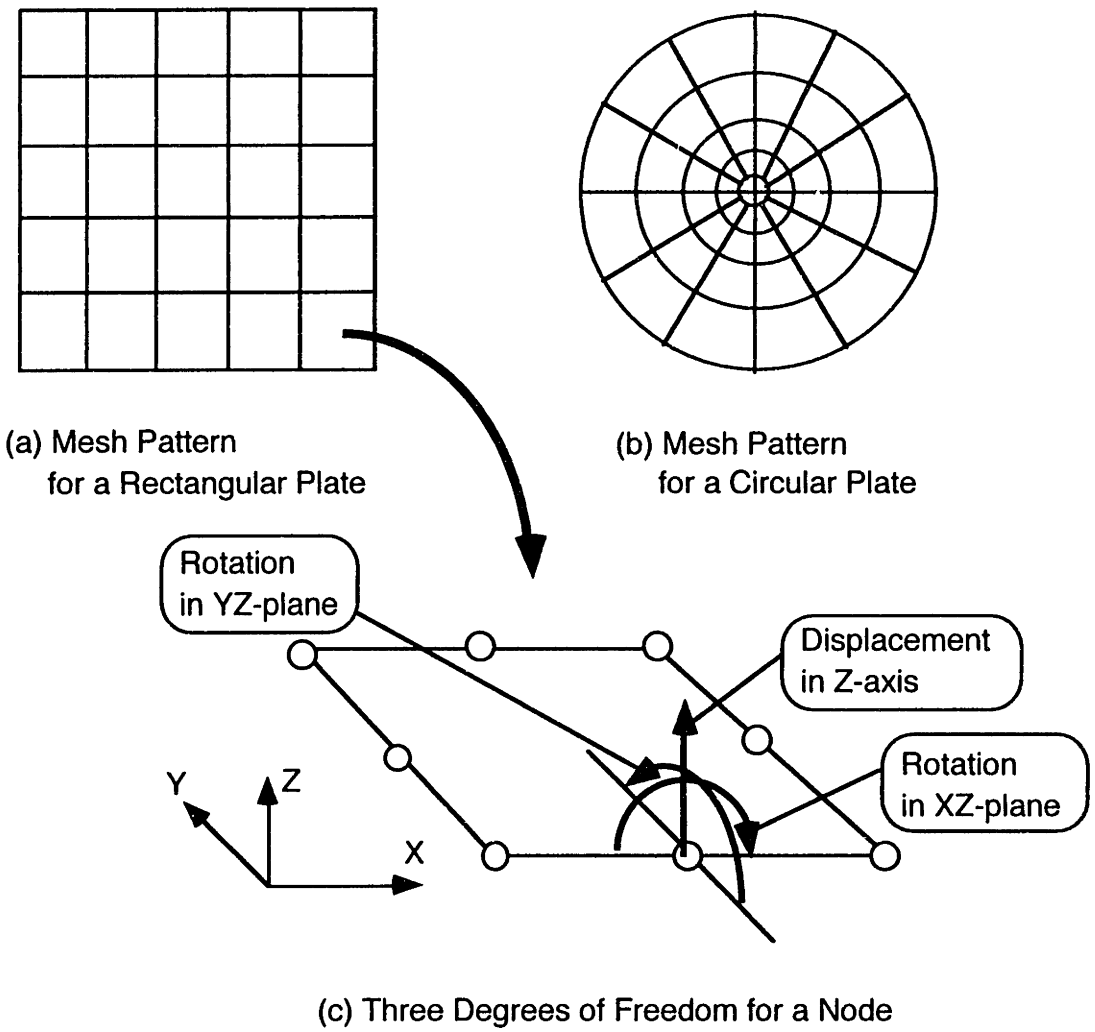


Figure 5-4: Mesh pattern and degrees of freedom for a node

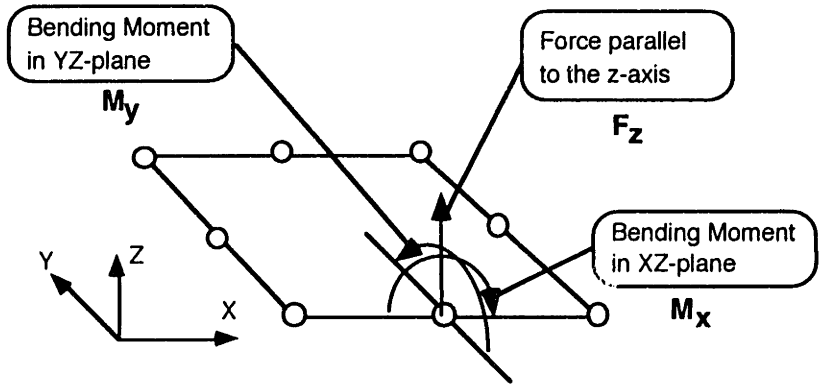


Figure 5-5: Loads for a node

Representation of a Heating Line

Although an actual final deformation is caused by plastic strains, a final deformation is obtained by applying a pair of bending moment rows in the linear FEM program. A heating line by a moving heat source onto a plate is represented with bending moments arranged in a pair of lines. Positive bending moments are arranged in a line parallel to a heating line in the distance of a and negative bending moments are arranged in another line parallel to the heating line at the same distance as shown in Figure 5-6(a).

Figure 5-6(b) shows accurate bending moment distribution according to the theory of discretization in FEM. When compared the result obtained by the accurate bending moment distribution with that by the evenly distributed bending moments, there was no significant difference in deformation. Therefore, evenly distributed bending moments are used to predict final shapes by the linear FEM program.

Determination of Amount of Bending Moments for Nodes

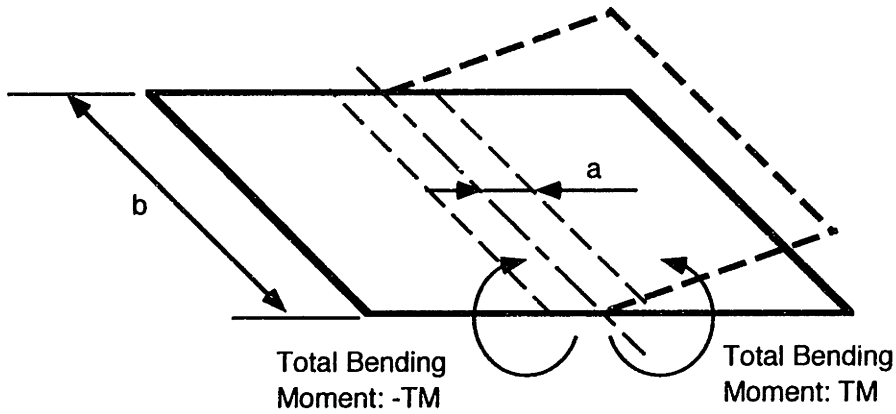
We need to determine the amount of the total bending moment due to bending moment rows arranged in the middle of a plate. The total bending moment due to a heating line is determined by a simple experiment of putting a heating line in the middle of a plate. The angular distortion is measured after the experiment and the total bending moment is calculated with the dimensions and Young's modulus of a plate by the following procedures.

The total bending moment that causes the half of the angular distortion of δ is expressed as follows using elastic theory.

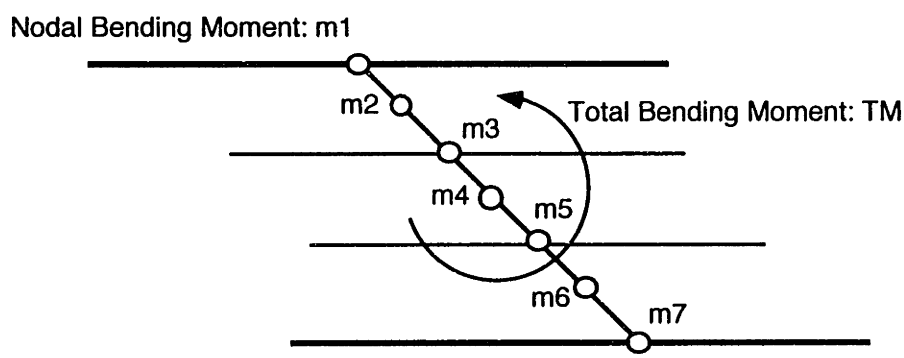
$$\delta = \arctan(\frac{TM \cdot a}{E \cdot I}), \quad (= \frac{\theta}{2}) \tag{5.2}$$

where TM is the total bending moment, E is Young's modulus, I is the sectional moment of inertia, and θ is the angular distortion.

The sectional moment of inertia of a plate, I, with a width of b and thickness of t is expressed as follows:



(a) Simulation Model of a Heating Line



m1:m2:m3:m4:m5:m6:m7=1:4:2:4:2:4:1
TM=m1+m2+m3+m4+m5+m6+m7

(b) Relation between Total Bending Moment
and Nodal Bending Moments

Figure 5-6: Simulation of a heating line

$$I = \frac{bt^3}{12}. \tag{5.3}$$

If we assume a small angular distortion,θ, can be expressed by the following equation:

$$\theta = 24 \cdot \frac{TM}{E \cdot t^3} \cdot \frac{a}{b}. \tag{5.4}$$

Finally, if we have the angular distortion of θ, the equivalent total bending moment is obtained by the following equation:

$$TM = \frac{\theta \cdot E \cdot t^3 \cdot b}{24 \cdot a}. \tag{5.5}$$

For example, if we use (θ = 5 degree = 5 $\times \pi/180$ radian, E = 196000 N/mm^2, t = 6.35 mm (0.25 inch), b = 300 mm, a = 30 mm, we get TM = 1,824,792 N-mm. Following the above mentioned procedure, we can determine the amount of the total bending moment due to a single heating line.

Next, we expand to the general case. Assuming the heat intensity and process speed are constant during a heat process, the total bending moment, TM, applied on the selected nodes is a function of the length of a heating line on a plate, LH. The total bending moment per unit length of the heating line, MPU, is considered as a constant and is determined by either experiments or simulation by thermal elastic-plastic FEM analysis. Then the total bending moment is expressed by the following equation:

$$TM = LH \times MPU. \tag{5.6}$$

Note that MPU is a function of the heating line position in the plate, the heat intensity of the heat process, and the process speed. The function must be found out with either series of experiments or thermal-elastic-plastic FEM calculations. At this moment, the function is unknown. It is necessary to obtain the relationship between MPU and the process parameters before the program is used in practice.

The total bending moment is a summation of the bending moments applied on nodes and it is shared equally among nodes. When we have n selected nodes, the bending moment applied on each node, M, is given as follows:

$$M = \frac{TM}{n}. \tag{5.7}$$

Positions of Heating Lines

In order to deal with arbitrary positions of heating lines including diagonal lines, the program must be able to select the nodes on which bending moments are applied. Also, the selected nodes should be classified either for positive bending moments or for negative ones. Then bending moments are applied on the nodes so that the total amount of the bending moments are equal to the total bending moment calculated from the length of a heating line.

Moreover, each bending moment for a node is decomposed into x-axis component and y-axis component so that the program can deal with diagonal heating lines.

A heating line is input as a set of two points, the start point and the end point of the heating line. Those two points must be located on the edges of a plate. As a result, the program can consider only passing-through heating lines on a plate and it cannot deal with heating lines stopping in the middle of a plate. So far, the program can deal with up to 20 heating lines.

Selection of Nodes to Apply Bending Moments

Nodes for applying bending moments are selected by the distances between a heating line and the nodes. (See Figure 5-7) Selection width, SW, is determined by the minimum distance between nodes so that only a row of nodes is selected when the heating line is parallel to either the x-axis or the y-axis.

The distance between a node and a heating line, d, is given by the following equation:

$$d = |\frac{-(y_2 - y_1)x_3 + (x_2 - x_1)y_3 + (y_2 - y_1)x_1 - (x_2 - x_1)y_1}{\sqrt{(y_2 - y_1)^2 + (x_2 - x_1)^2}}|, \tag{5.8}$$

where (x_1, y_1) is the coordinates of a point on the heating line, (x_2, y_2) is the coordinates of another point on the heating line, (x_3, y_3) is the coordinates of the node.

Separate the numerator inside the absolute operator in the above equation as nd.

$$nd = -(y_2 - y_1)x_3 + (x_2 - x_1)y_3 + (y_2 - y_1)x_1 - (x_2 - x_1)y_1. \tag{5.9}$$

We can classify the side of a node (whether the upper side or the lower side of the heating line in Figure 5-7) by the sign of nd. In other words, the sign of the bending moment applied on a node can be determined with the sign of nd. For example, if $nd > 0$ then positive bending moments are applied on nodes or if $nd < 0$ then negative moments are applied.

Applying Bending Moments on Nodes

When a heating line is not parallel to the x-axis or the y-axis, the bending moment on a node has to be given as the combination of the bending moments in xz-plane and yz-plane, M_x, M_y respectively. Suppose that the angle between a heating line and the x-axis is $\alpha(-\frac{\pi}{2} < \alpha \leq \frac{\pi}{2})$, M_x and M_y are given the following equations respectively:

If $nd > 0$, then

$$M_x = M \cdot \sin\alpha, \quad M_y = -M \cdot \cos\alpha. \tag{5.10}$$

If $nd \leq 0$, then

$$M_x = -M \cdot \sin\alpha, \quad M_y = M \cdot \cos\alpha. \tag{5.11}$$

Also, the angle $\alpha(-\frac{\pi}{2} < \alpha \leq \frac{\pi}{2})$ is given by the following equation using the coordinates of two points on the heating line:

$$\alpha = \arctan(-\frac{y_2 - y_1}{x_2 - x_1}), \tag{5.12}$$

where (x_1, y_1) is the coordinates of a point on the heating line, (x_2, y_2) is the coordi-

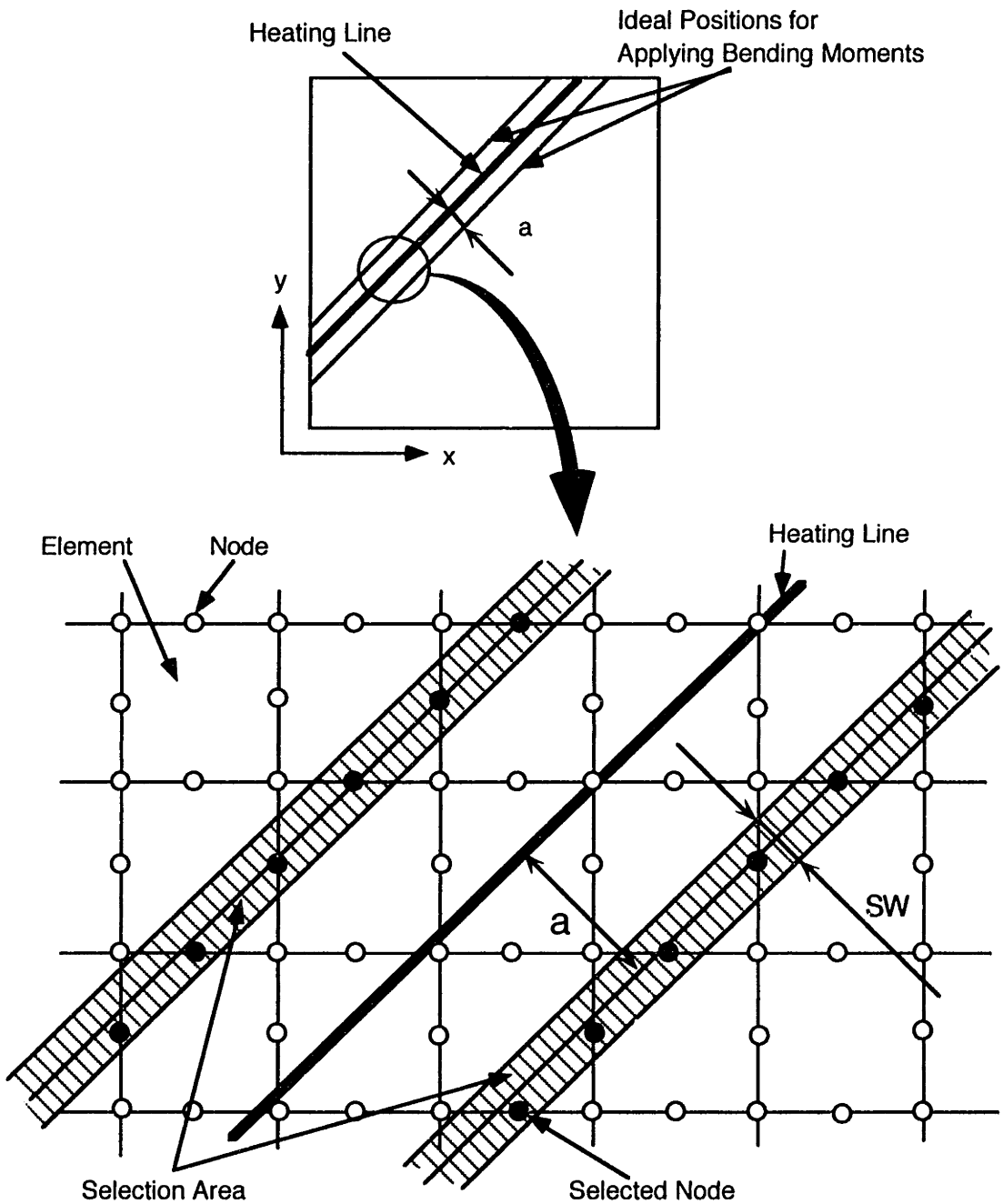


Figure 5-7: Selection of nodes for applying bending moments

nates of another point on the heating line.

5.3.5 Representation of the Load Matrix

These bending moments have to be applied on a node as a function of the magnitudes of heating lines in order to make the iteration of calculations fast.

The bending moments applied to nodes are stored as the elements of two dimensional matrix $ELOAD(i, k)$ in the program. Letter i shows the ith element on the plate and letter k shows the kth degree of freedom in the element. (One element has 24 degrees of freedom.) Suppose we have the magnitudes of heating lines, Q_1, Q_2, Q_3, ... These values are given by the GA driver as a set of heating conditions. Therefore, the magnitudes should be dealt with as unknown values in the program. Otherwise, the program needs much calculation to obtain the elements of $ELOAD$. This causes long calculation time for many iterations needed for genetic algorithm. On the other hand, if the elements of $ELOAD$ are expressed as the following expression, the calculation time is significantly reduced:

$$ELOAD(i, k) = \sum_{l=1}^{n} S(i, k, l) \cdot Q_l, \tag{5.13}$$

where $S(i, k, l)$ is the coefficient of Q_l for the k th degree of freedom in the i th element, and n is the number of heating lines.

Since $S(i, k, l)$ can be calculated when the positions of heating lines are given and its value never changes during the iterative calculation for genetic algorithm, the above formation of the load matrix reduces calculation time.

5.4 Summary

The program that can be used for practical uses has been implemented. It uses a linear FEM program as a decoding part and a GA program as an optimizing part. In the linear FEM program, a heating line on a plate are represented by a pair of rows of bending moments. The intensities of heat processes are encoded in the GA

part for optimization procedures. The program can determine the heating conditions for arbitrary passing-through straight heating lines on a plate. However, since the decoding FEM program is based on the thin plate theory, it assumes the linearity of addition of plastic strains, and it ignores inplane shrinkage of a plate, the program should be used when the plate is nearly flat during the process.

Chapter 6

Validation of the Program

The validation of the program consists of two parts: the validation of the decoding program and the validation of the optimizing program. The validation of the decoding program is comparison of experimental final shapes from laser forming with the shapes calculated by the program. The validation of the optimizing program is the search results for a complicated shape whose heat process conditions are known. Also, a trial has been done using an experimental final shape from laser forming as a target shape in the validation of the optimizing program.

6.1 Validation of the Decoding Program

In this section, the final shapes due to laser forming are compared to those calculated by the decoding program.

6.1.1 One Heating Line in the Middle of a Plate

In this section, the calculation result with the FEM decoding program and an experimental result processed by a CO_2 laser beam have been compared in terms of deformation. A heating line parallel to the y-axis was put in the middle of a square plate during the process. The amount of total bending moment in the FEM calculation conditions was determined to match the angular distortions in the experimental

data and the calculation.

FEM calculation conditions were as follows:

- E=196000N/mm^2,
- ν=0.3,
- Dimensions=300mm (in x-axis) $\times$ 300mm (in y-axis),
- Thickness=6.35mm,
- Number of Elements=25,
- Number of Nodes=96,
- Size of Elements=60mm $\times$ 60mm,
- Total Bending Moment for a Line=1824790N-mm,
- Position of Heating Line: x=150mm.

An experiment was conducted under the following conditions:

- Material: Inconel 625,
- Dimensions=304.8mm (x-axis) $\times$ 304.8mm (y-axis),
- Thickness=6.35mm,
- Laser Beam: CO_2 Laser Beam,
- Output of a Laser Beam=8.4kW,
- Spot Size of a Laser Beam=60mm,
- Process Speed=6.67mm/sec,
- Position of Heating Line: x=150mm.

The deformation calculated by the FEM decoding program is shown in Figure 6-1. Figure 6-2 shows the experimental result measured by a coordinate measuring machine (CMM). The figures show the deformed mesh, along with contour lines of constant displacement. (By hindsight, it would have been better to use different line weights for the two.)

Angular distortions were calculated along the y-axis in both results. The comparison in angular distortion is shown in Figure 6-3. Since the total bending moment in the FEM calculation has been adjusted to match the experimental angular distortion, the amounts of angular distortion along the y-axis are very similar to each other. One different point among the data is the increase of angular distortion from the start point to the end point of the process in the experiment. This increase cannot be expressed in the result of the FEM decoding program.

Profiles perpendicular to the heating line are plotted in Figure 6-4. In the figure, open symbols show the profiles at the edge of the plate and solid symbols show the profiles in the middle of the plate. The profiles of the calculation and the experiment shows good agreement at both places.

The above mentioned comparison results show that the calculated deformation is close to the experimental result for a process of one heating line in the middle of a plate, with 6mm deflection in a 200 - 300mm plate 6mm thick.

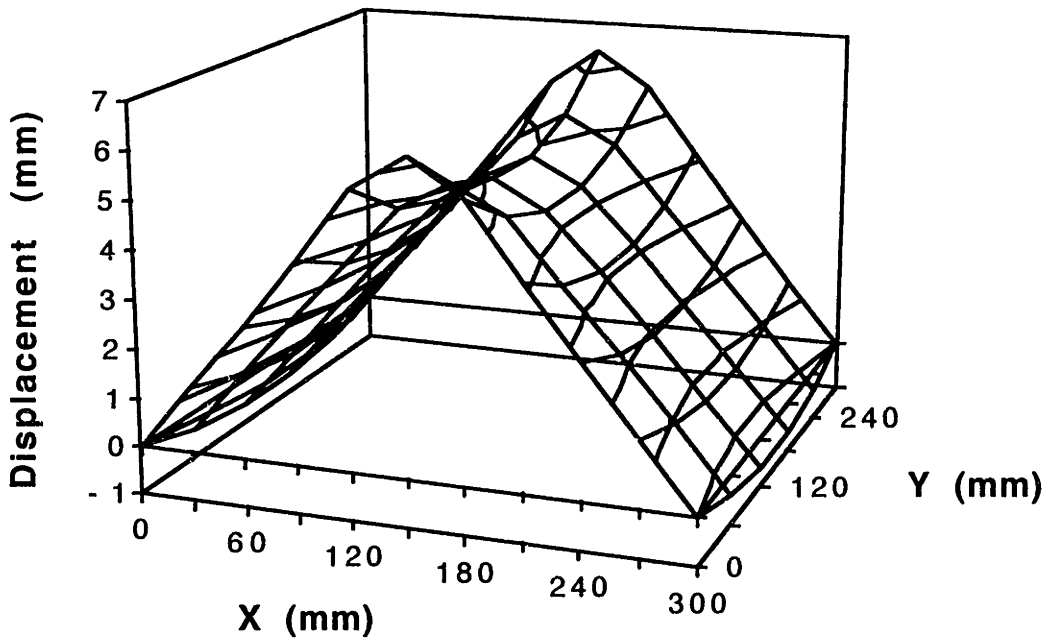


Figure 6-1: An FEM calculation result for one heating line in the middle of a plate

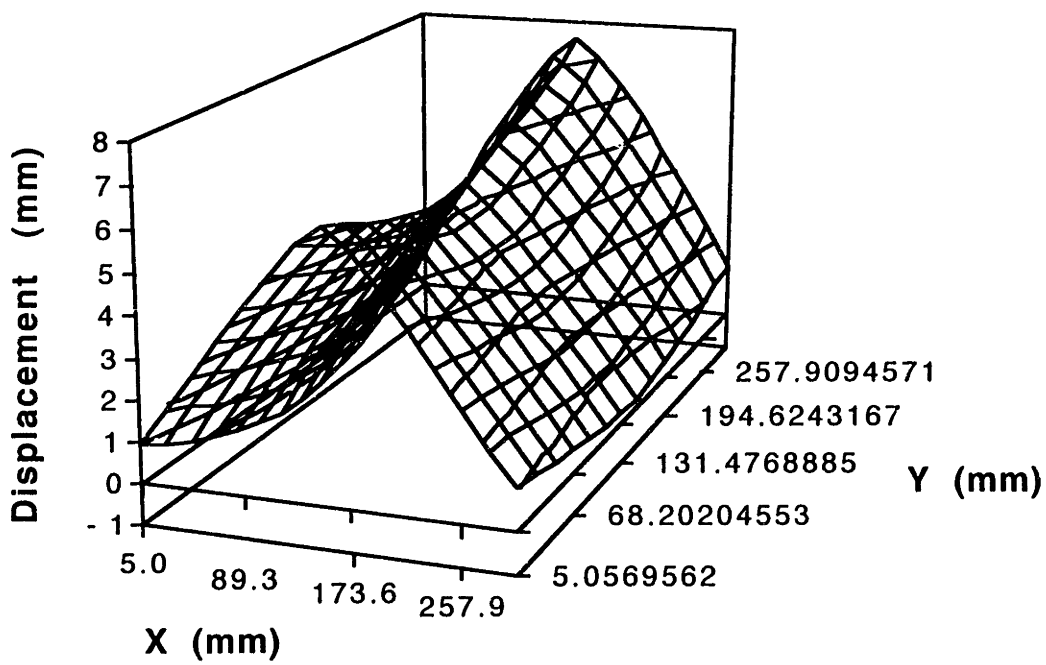


Figure 6-2: An experimental final shape for one heating line in the middle of a plate of Inconel 625

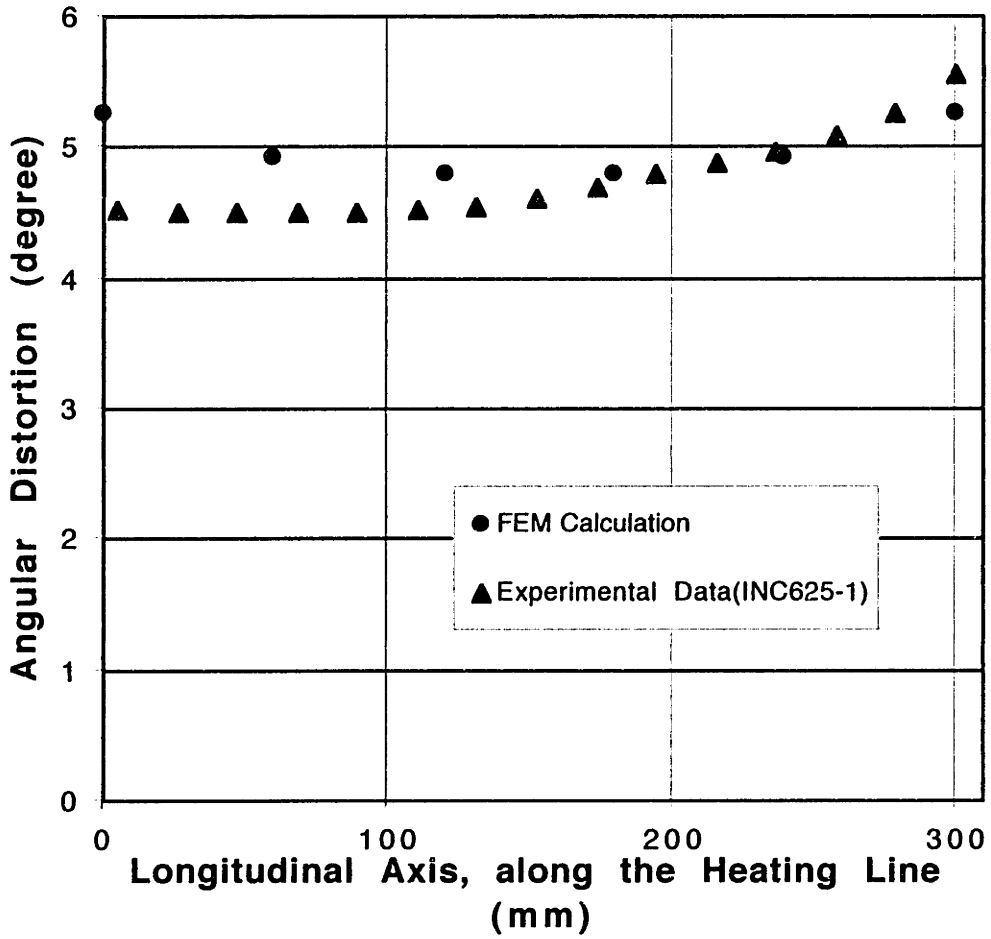


Figure 6-3: Comparison in angular distortion between the experimental final shape and the FEM calculation result for one heating line

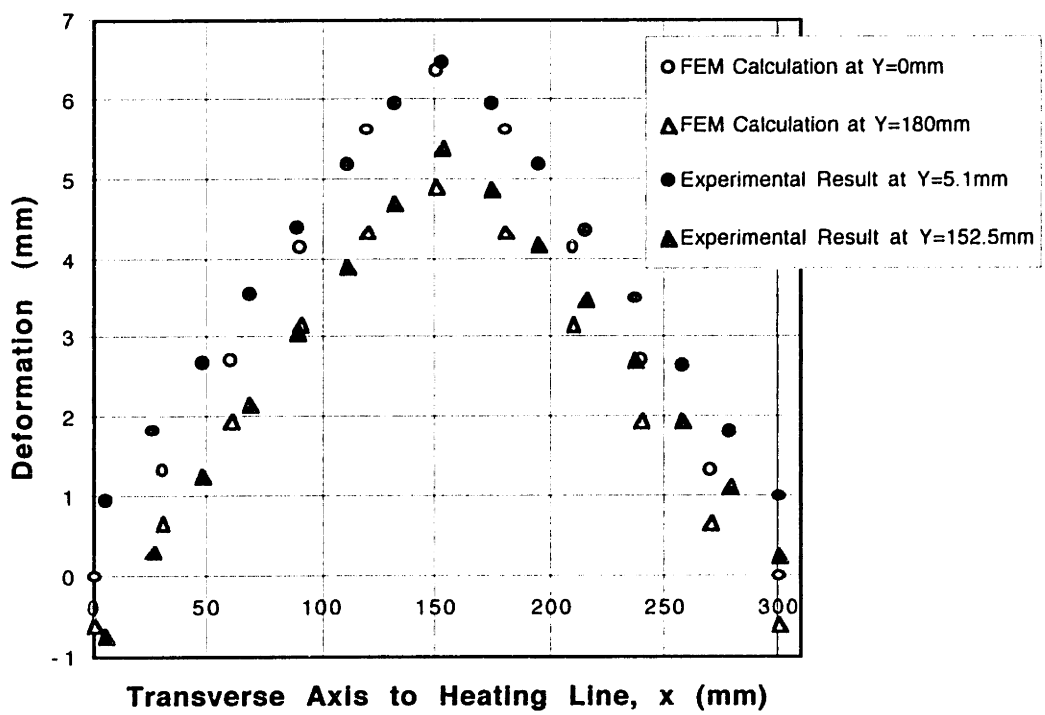


Figure 6-4: Comparison in profiles between the experimental final shape and the FEM calculation result for one heating line

6.1.2 Two Heating Lines in the Middle of a Plate Crossing Each Other

In this section, the deformation calculated by the FEM decoding program is compared to the experimental result for the two heating line process. Two heating lines crossing each other at the center of a rectangular plate have been processed. The total bending moment in the FEM calculation conditions was determined to match the deformation calculated by the FEM program to the experimental shape.

FEM calculation conditions were as follows:

- E=196000N/mm^2,
- ν=0.3,
- Dimensions=300mm (in x-axis) $\times$ 230mm (in y-axis),
- Thickness=9mm,
- Number of Elements=25,
- Number of Nodes=96,
- Size of Elements=60.0mm $\times$ 38.33mm,
- Total Bending Moment for a Line=330000N-mm,
- Separation Length of Bending Moment (a) =30mm,
- Positions of Heating Line: x=150mm and y=115mm.

The experiment was conducted under the following conditions:

- Material: HSLA-100,
- Dimensions=304.8mm (in x-axis) $\times$ 228.6mm (in y-axis),
- Thickness=9mm,
- Laser Beam: Nd:YAG Laser Beam,

- Output of a Laser Beam=1.8kW,
- Spot Size of a Laser Beam=6.35mm,
- Oscillation Width=25.4mm,
- Oscillation Rate=13Hz,
- Process Speed=8.467mm/sec,
- Positions of Heating Line: x=152.4mm and y=124.3mm.

The shape of a plate calculated by the FEM program is shown in Figure 6-5. The experimental final shape processed by an Nd:YAG laser beam is shown in Figure 6-6.

Figure 6-7 shows the comparison of the profiles parallel to the x-axis. Figure 6-8 shows the comparison of the profiles parallel to the y-axis. The comparison in Figure 6-7 shows the complete agreement between two shapes both at the edge of the plate and in the middle. The comparison in Figure 6-8 shows agreement within 0.15mm out of 2.5mm at the center; and 0.3mm out of 1.5mm at the edge.

Although there is a slight difference in the profiles parallel to the y-axis at the edge, the calculated deformation is close enough to the experimental result.

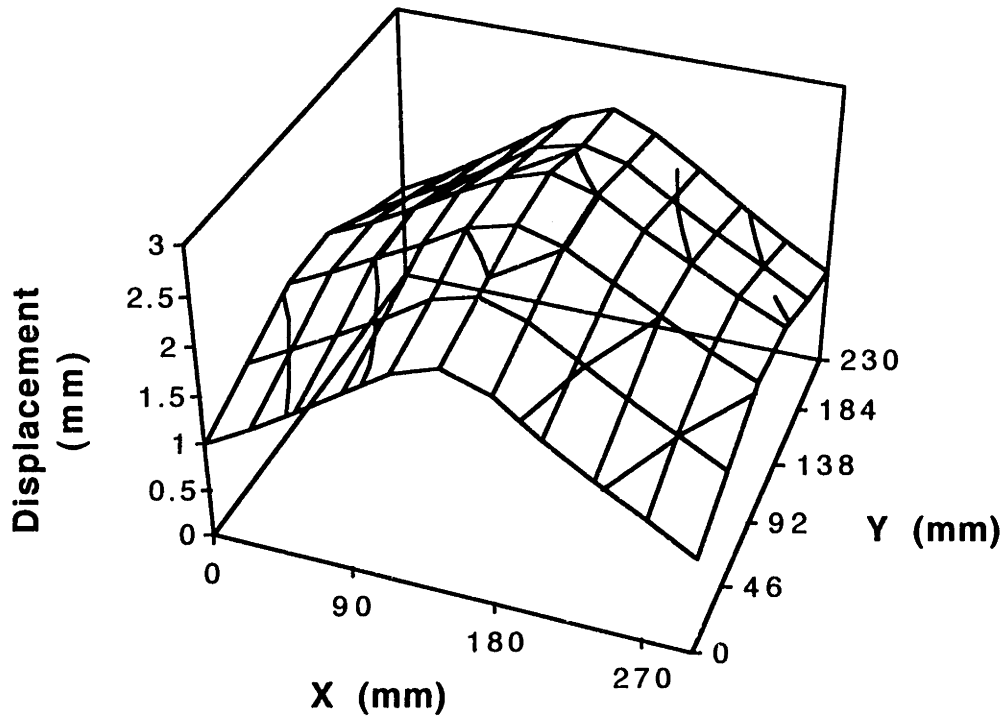


Figure 6-5: An FEM calculation result for two heating lines crossing each other at the center of a plate

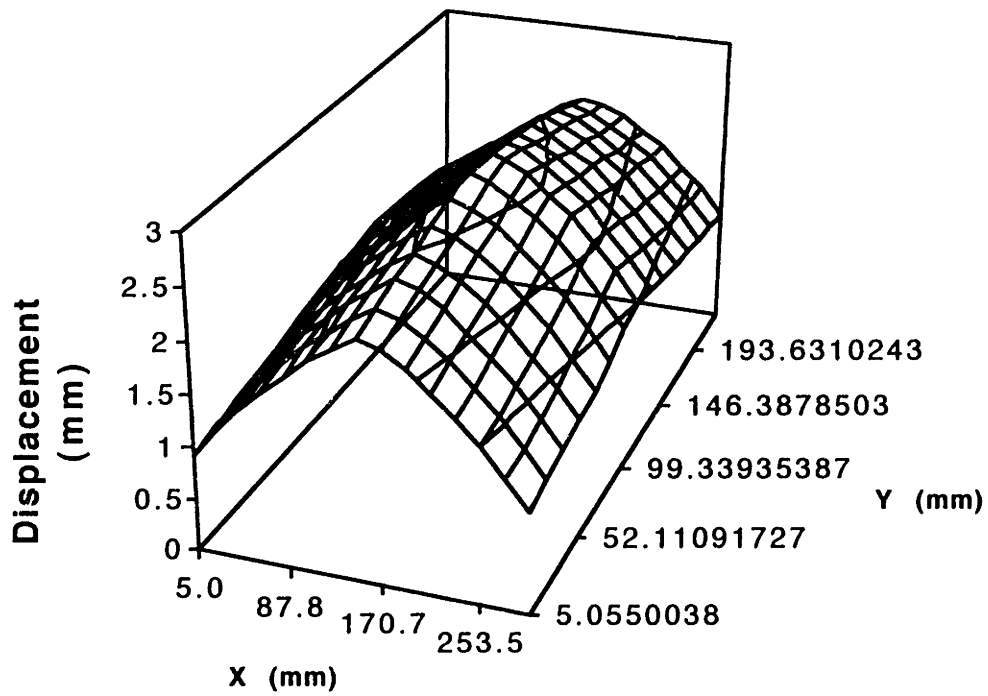


Figure 6-6: An experimental final shape for two heating lines crossing each other at the center of a plate of HSLA-100

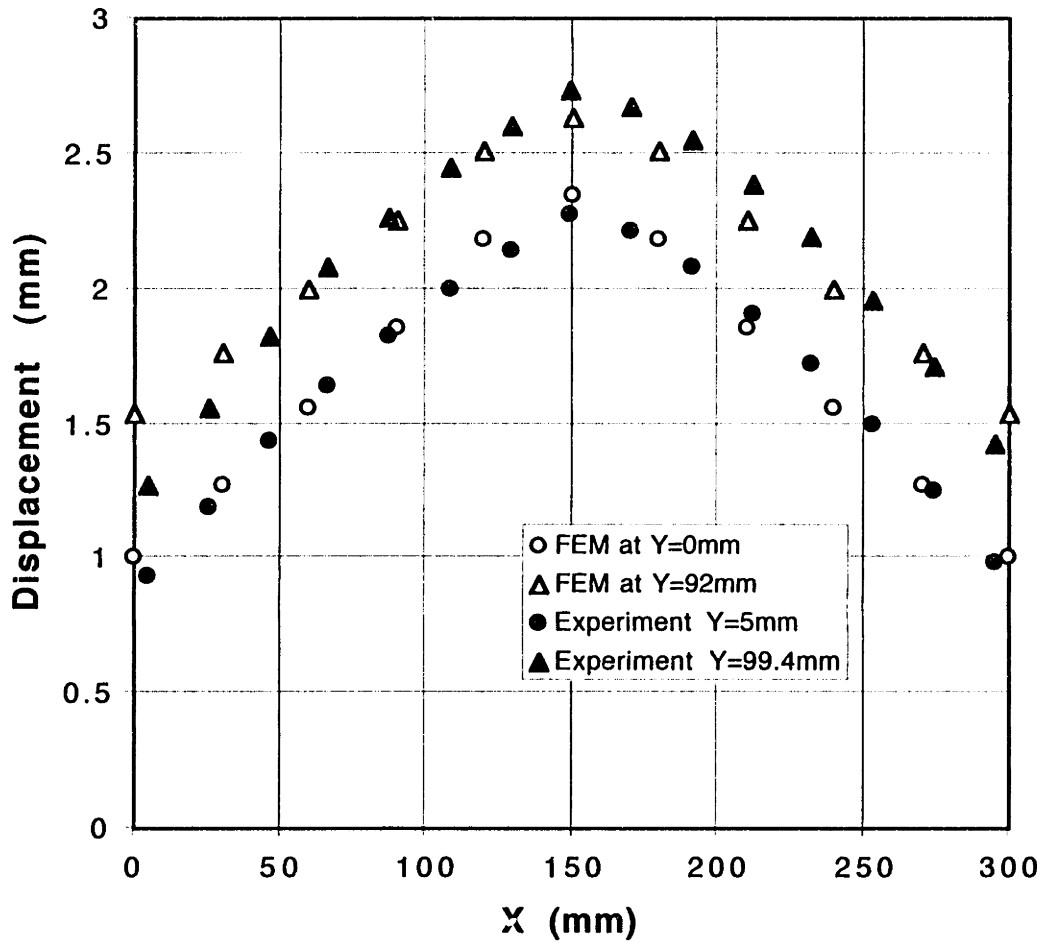


Figure 6-7: Comparison in profiles parallel to the x-axis between the experimental final shape and the FEM calculation result for two heating lines crossing each other at the center of a plate

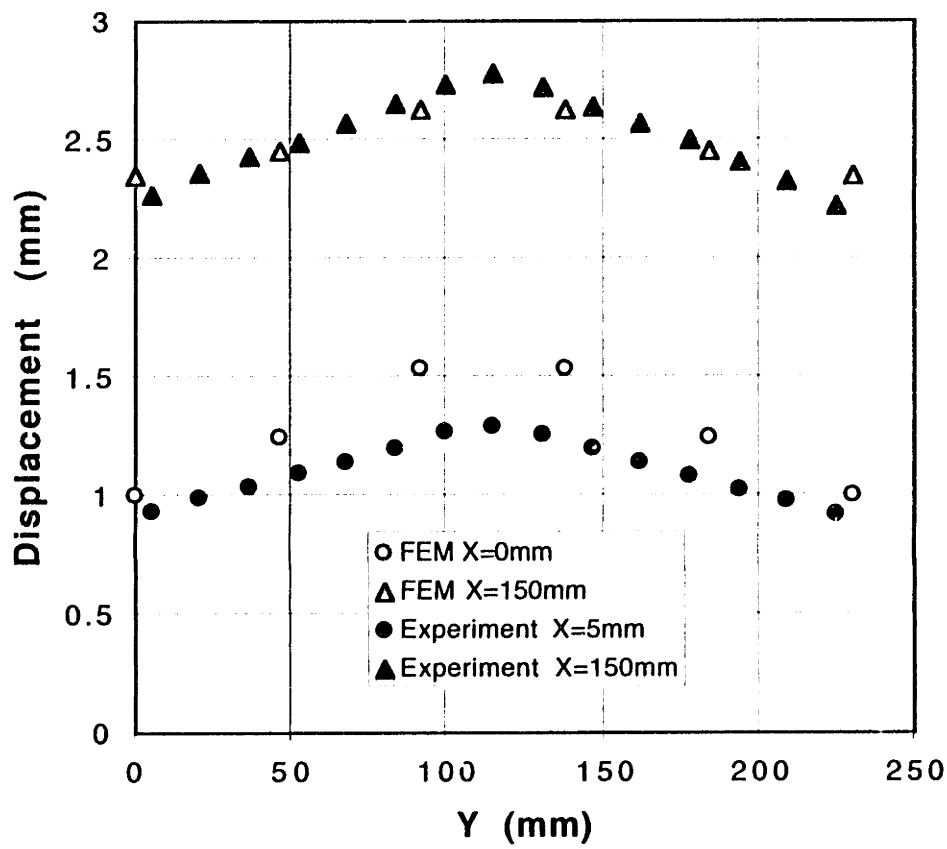


Figure 6-8: Comparison in profiles parallel to the y-axis between the experimental final shape and the FEM calculation result for two heating lines crossing each other at the center of a plate

6.1.3 Multi Heating Lines

Three experimental results are compared to the results calculated by the decoding FEM program for multi heating lines.

Multi Heating Line Case 1

The heating pattern for Case 1 is shown in Figure 6-9. The numbers next to the lines in the figure show the order of the processes and the arrows show the direction of the processes.

The calculation has been carried out under the following conditions:

- E=196000N/mm^2,
- ν=0.3,
- Dimensions=300mm $\times$ 300mm,
- Thickness=6.35mm,
- Number of Elements=144,
- Number of Nodes=481,
- Size of Elements=25.0 $\times$ 25.0mm,
- Bending Moment per Unit Length for 1,5,and 6: 6122 N-mm/mm,
- Bending Moment per Unit Length for 2,3,and 4: 7951 N-mm/mm,
- Separation Length of Bending Moment (a) =12.5mm.

The experiment was conducted under the following conditions:

- Material: Inconel 625,
- Dimensions=304.8mm $\times$ 304.8mm,
- Thickness=6.35mm,

- Laser Beam: Nd:YAG Laser Beam,
- Output of a Laser Beam=2.5kW,
- Spot Size of a Laser Beam=20mm,
- Process Speed for Lines 1, 5, and 6: 5.0mm/sec,
- Process Speed for Lines 2, 3, and 4: 3.33mm/sec.

Since the laser beam was traversed at two different speeds on the plate, the bending moment per unit length has two values. The ratio of the values of the bending moment per unit length, MPU, was determined from:

$$MPU \sim \frac{P}{t\sqrt{v}} \tag{6.1}$$

where P is power of a laser beam, t is thickness of a plate, v is process speed.

The shape of a plate calculated by the FEM program is shown in Figure 6-10. The experimental final shape processed by an Nd:YAG laser beam is shown in Figure 6-11.

Figure 6-12 shows the comparison of the profiles parallel to the x-axis. Figure 6-13 shows the comparison of the profiles parallel to the y-axis. The comparison in Figure 6-12 shows agreement within 1mm out of 8mm at the center and only within 1mm out of 3mm at some other points. The comparison in Figure 6-13 shows good agreement in the middle of the plate but only up to 2mm out of 5mm at some other points.

In Case 1, although there is a difference in the profiles parallel to the y-axis at the edge, the calculated deformation is close enough to the experimental result.

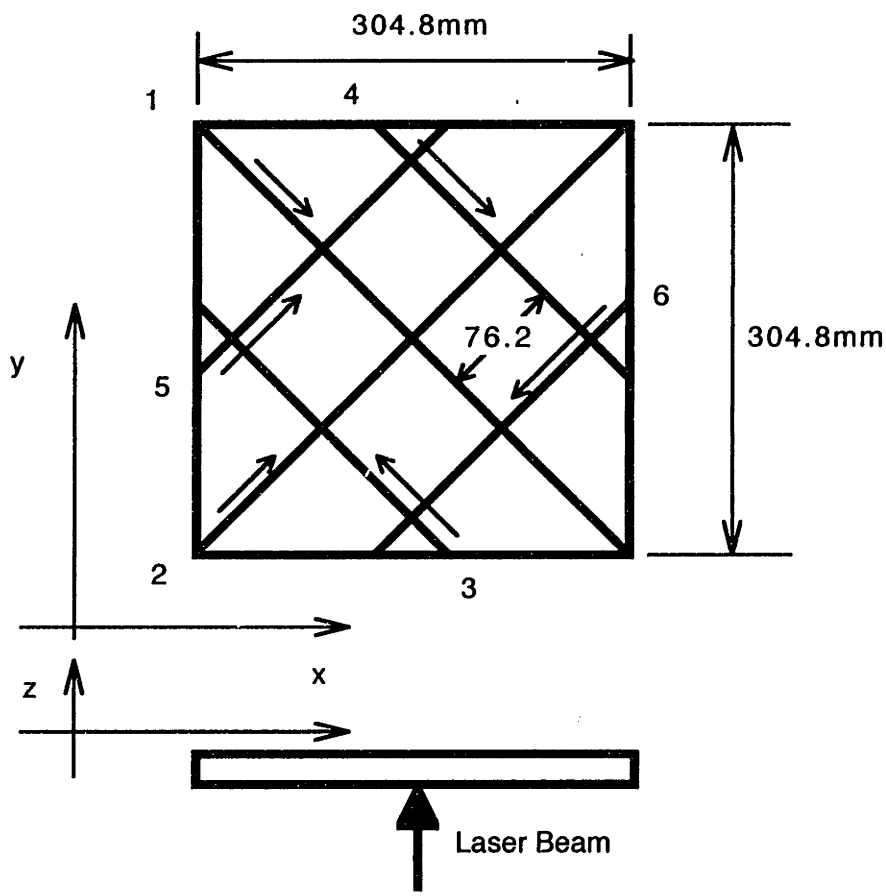


Figure 6-9: Heating pattern for multi heating lines Case 1 in Inconel 625

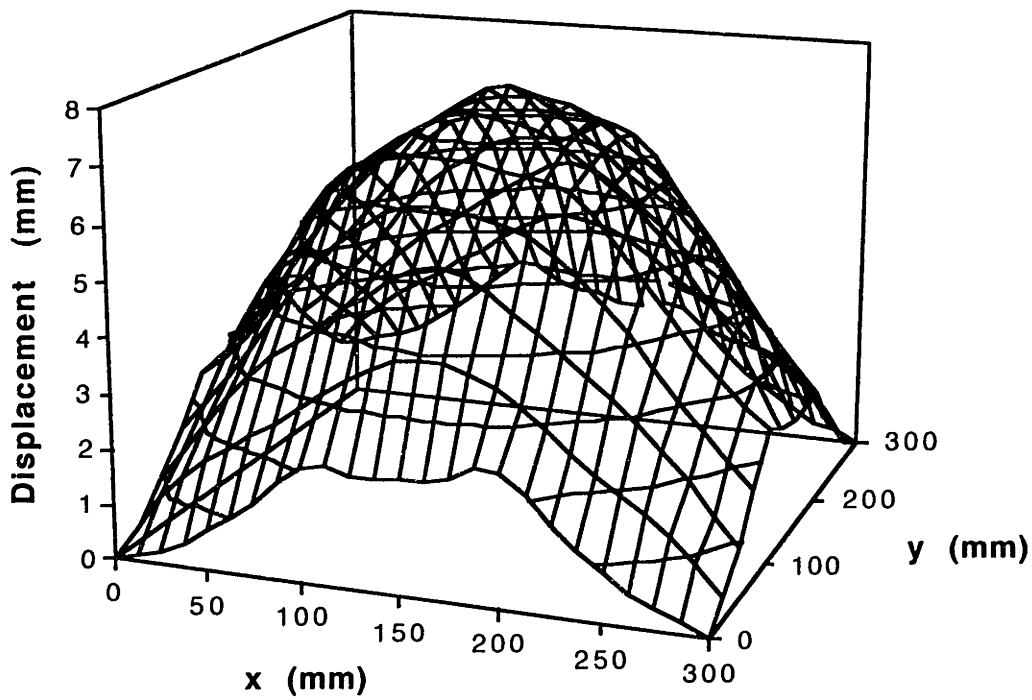


Figure 6-10: FEM calculation result for multi heating lines Case 1

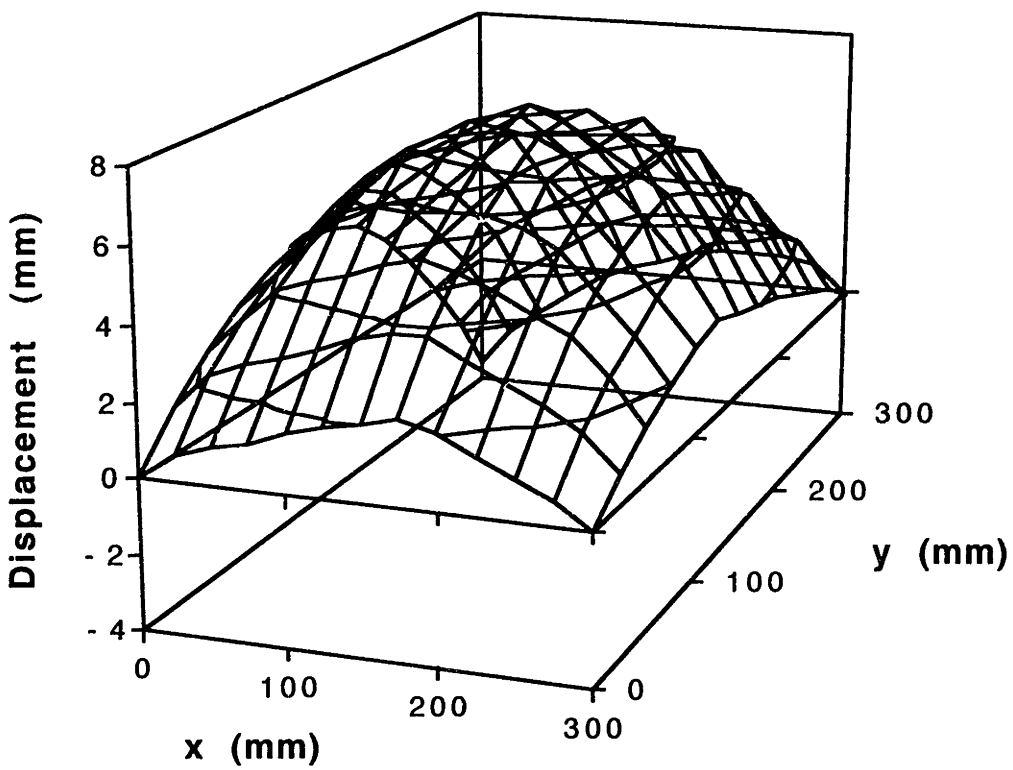


Figure 6-11: Experimental result for multi heating lines Case 1

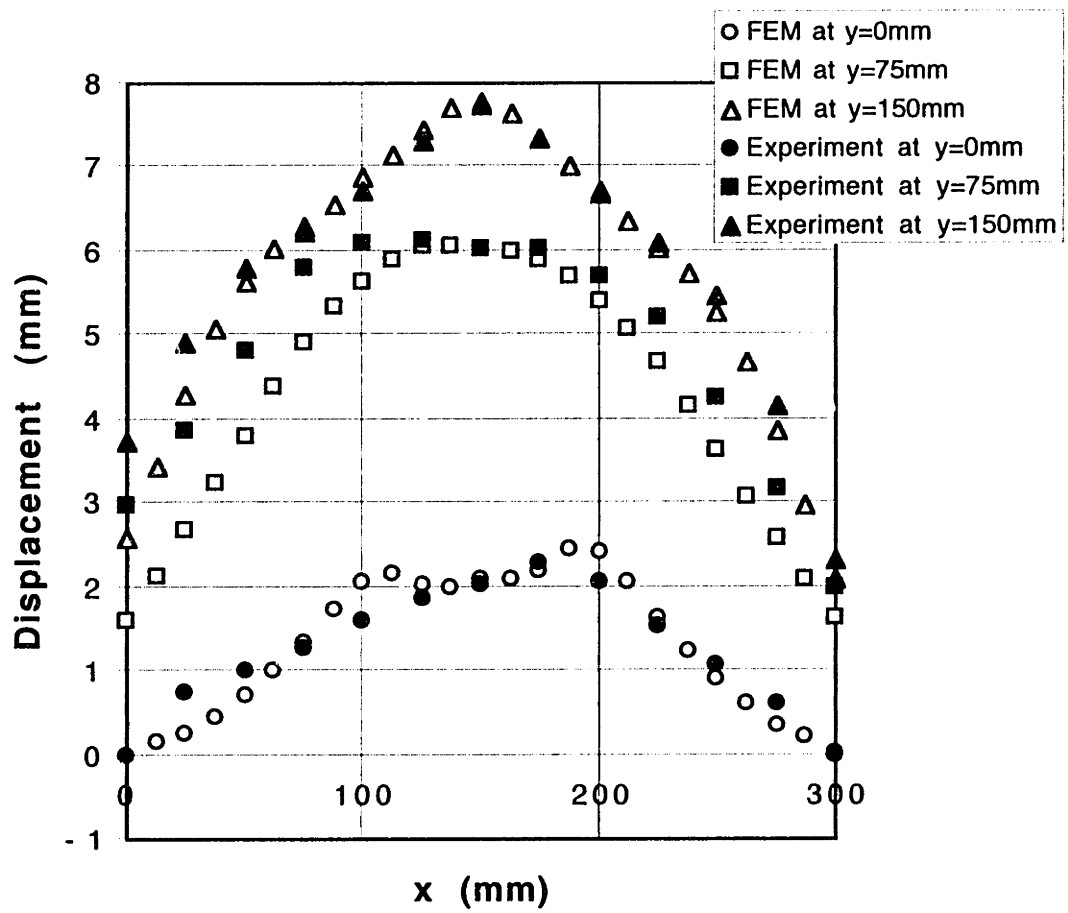


Figure 6-12: Comparison in profiles parallel to the x-axis between the FEM calculation result and the experimental result for multi heating lines Case 1

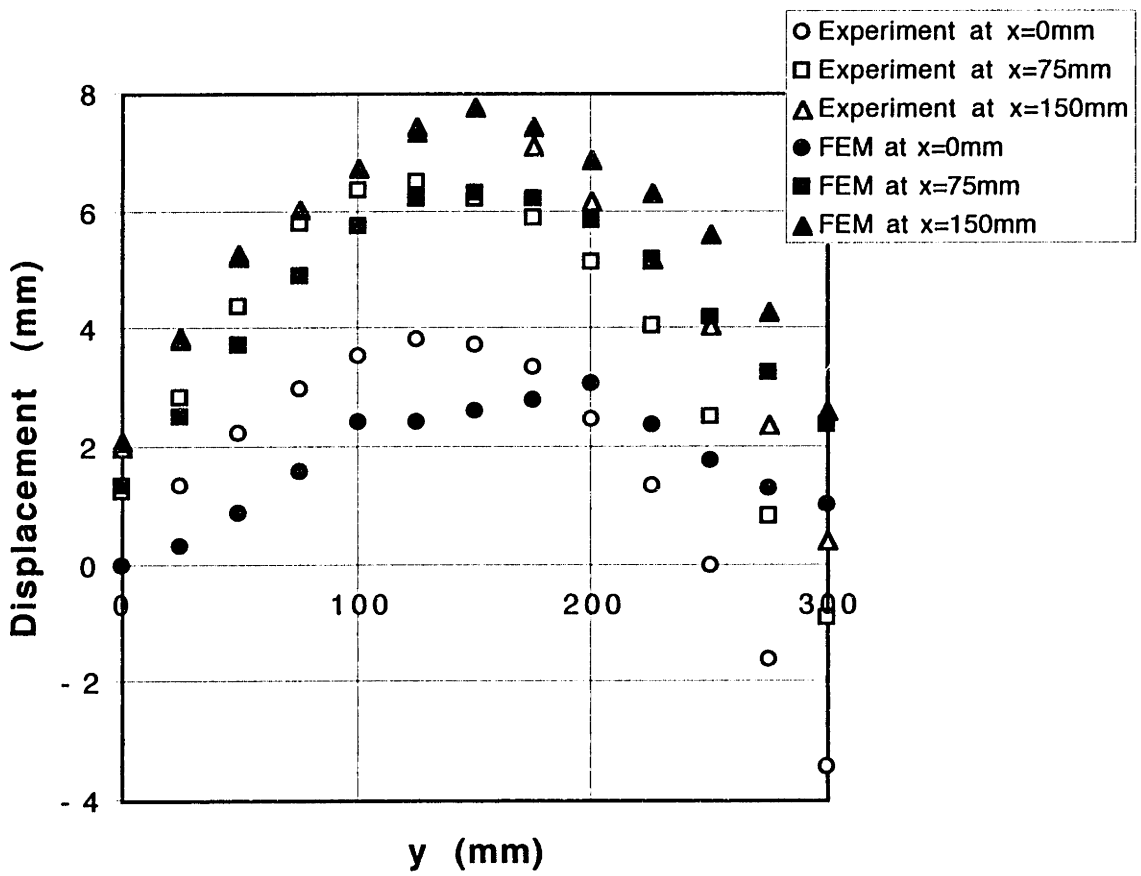


Figure 6-13: Comparison in profiles parallel to the y-axis between the FEM calculation result and the experimental result for multi heating lines Case 1

Multi Heating Line Case 2

The heating pattern for Case 2 is shown in Figure 6-14. The numbers next to the lines in the figure show the order of the processes. The heating lines were applied to the plate in a grid with pitch of 50.8mm.

The calculation has been carried out under the following conditions:

- E=196000N/mm^2,
- ν=0.3,
- Dimensions=300mm $\times$ 300mm,
- Thickness=6.35mm,
- Number of Elements=144,
- Number of Nodes=481,
- Size of Elements=25.0 $\times$ 25.0mm,
- Bending Moment per Unit Length for Each Heating Line: 7591 N-mm/mm,
- Separation Length of Bending Moment (a) =12.5mm.

The experiment was conducted under the following conditions:

- Material: HSLA-80,
- Dimensions=304.8mm $\times$ 304.8mm,
- Thickness=6.35mm,
- Laser Beam: Nd:YAG Laser Beam,
- Output of a Laser Beam=2.5kW,
- Spot Size of a Laser Beam=20mm,
- Process Speed=3.33mm/sec.

The shape of a plate calculated by the FEM program is shown in Figure 6-15. The experimental final shape processed by an Nd:YAG laser beam is shown in Figure 6-16.

Figure 6-17 shows the comparison of the profiles parallel to the x-axis. Figure 6-18 shows the comparison of the profiles parallel to the y-axis. The profiles parallel to the x-axis show relatively good agreement between the FEM calculation and the experimental result. However, the profiles parallel to the y-axis in the experimental result are totally different from those in the calculation. In Figure 6-18, the three profiles in the experimental result are almost straight. These profiles cannot be simulated by the FEM calculation. Also, deformation along one edge is very asymmetrical.

Although the heating lines were processed carefully to avoid the effect of the previous shape, that is from the center to the edge and the heating line parallel to the y-axis and that parallel to the x-axis alternately, heating lines parallel to the x-axis do not seem to bend the plate. This phenomenon is explained by the effect of the initial shape of a plate on the bending shape by a heating line. In other words, since the plate stiffness to the second heating line was increased by the deformation due to the first heating line, the second heating line could not bend the plate as much as it was supposed to do. The third heating line could effectively bend the plate since its stiffness to the third heating line had not increased due to the second heating line. In short, the first heating line has determined the direction of the dominant curvature.

On the other hand, in the two heating line case presented in the previous section, the FEM calculation matched to the experimental result. The difference between the two heating line case and the multi heating line Case 2 is the intensity of the heat process. In the two heating line case, the deformation due to the first heating line is small enough not to affect the deformation due to the second heating line. In the multi heating line Case 2, the deformation due to the first heating line seemed to be large enough to reduce the deformation due to the second heating line.

The result of the multi heating line Case 2 shows that we need to consider the order of the heat processes when the deformation due to a heating line is large. So far, the FEM decoding program cannot deal with the heating processes that produce deformations large enough to change the stiffness of a plate.

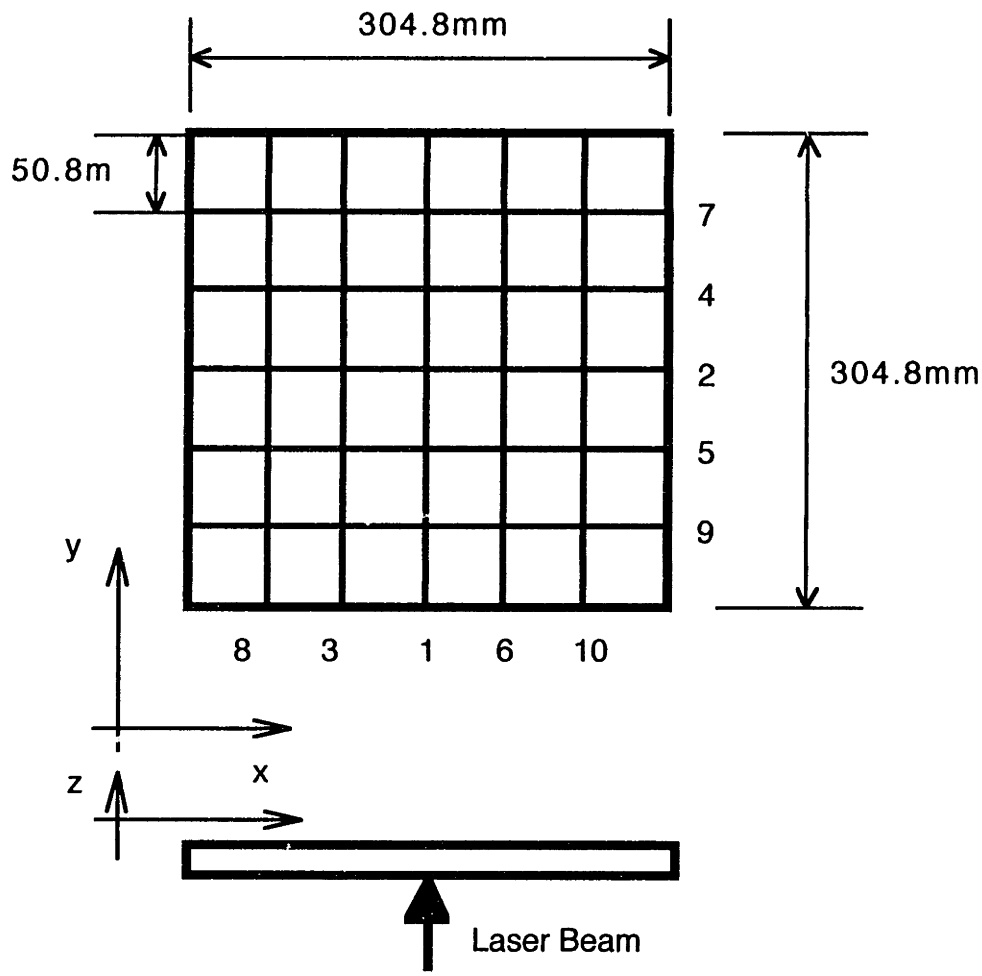


Figure 6-14: Heating pattern for multi heating lines Case 2 in HSLA-80

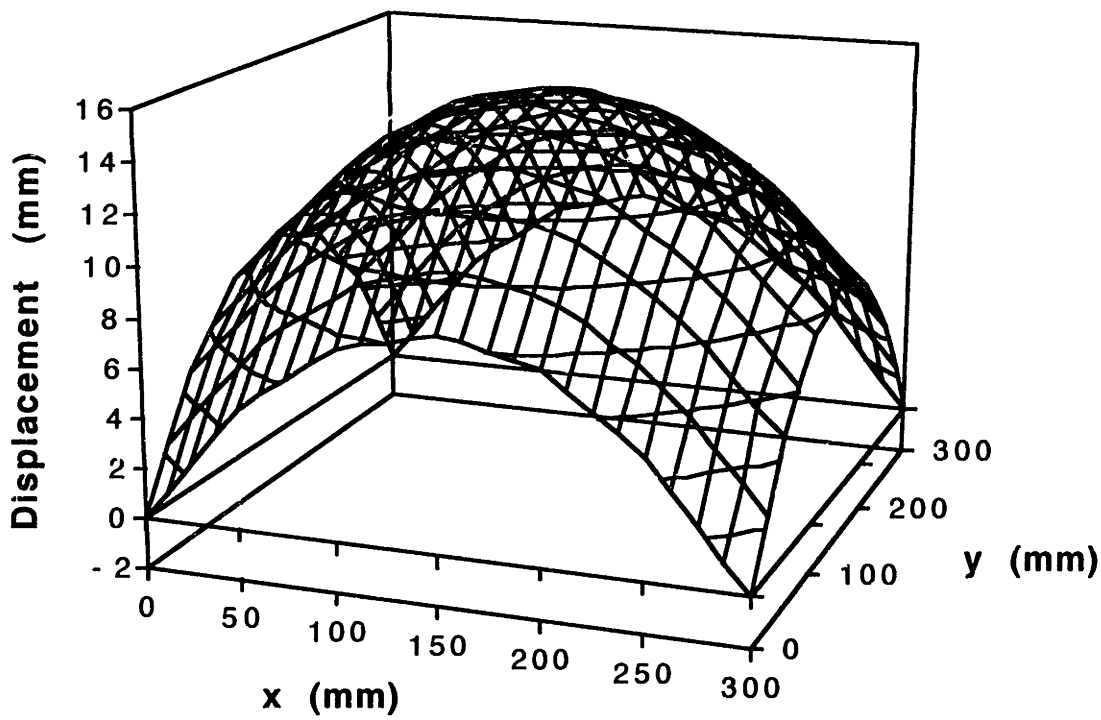


Figure 6-15: FEM calculation result for multi heating lines Case 2

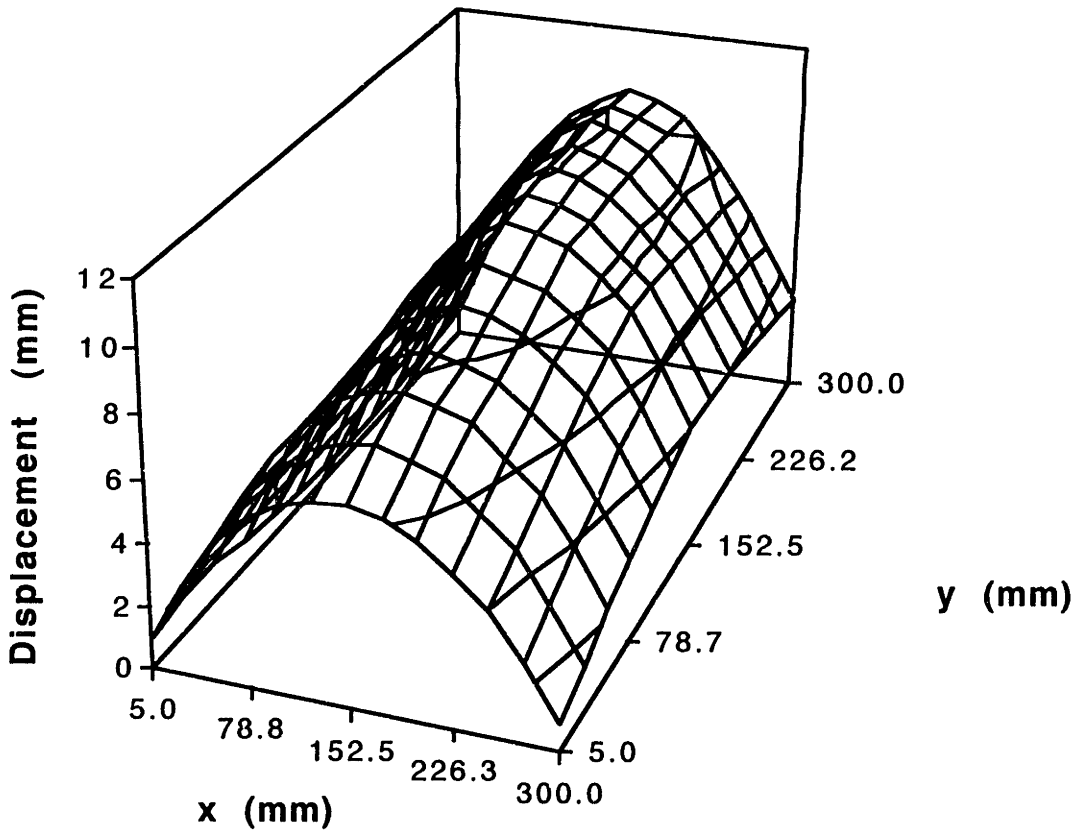


Figure 6-16: Experimental result for multi heating lines Case 2

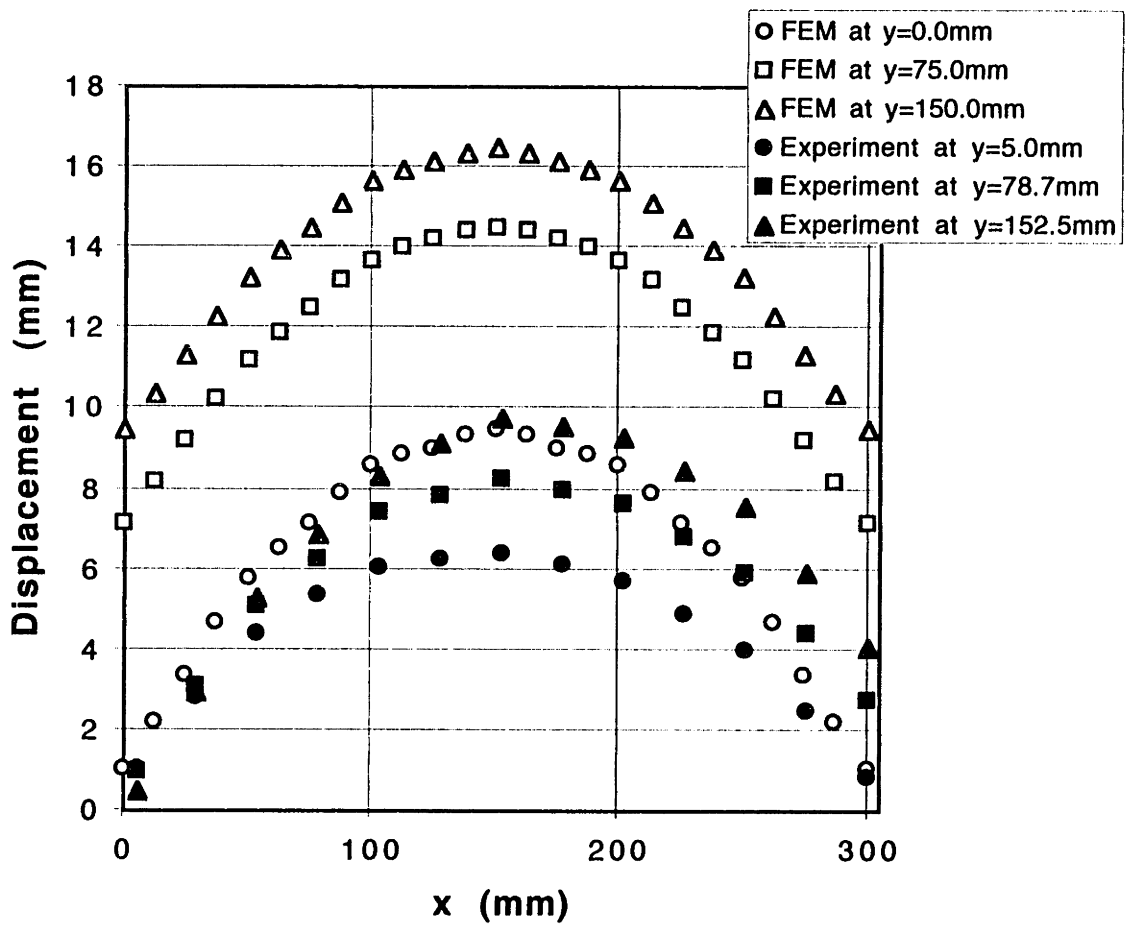


Figure 6-17: Comparison in profiles parallel to the x-axis between the FEM calculation result and the experimental result for multi heating lines Case 2

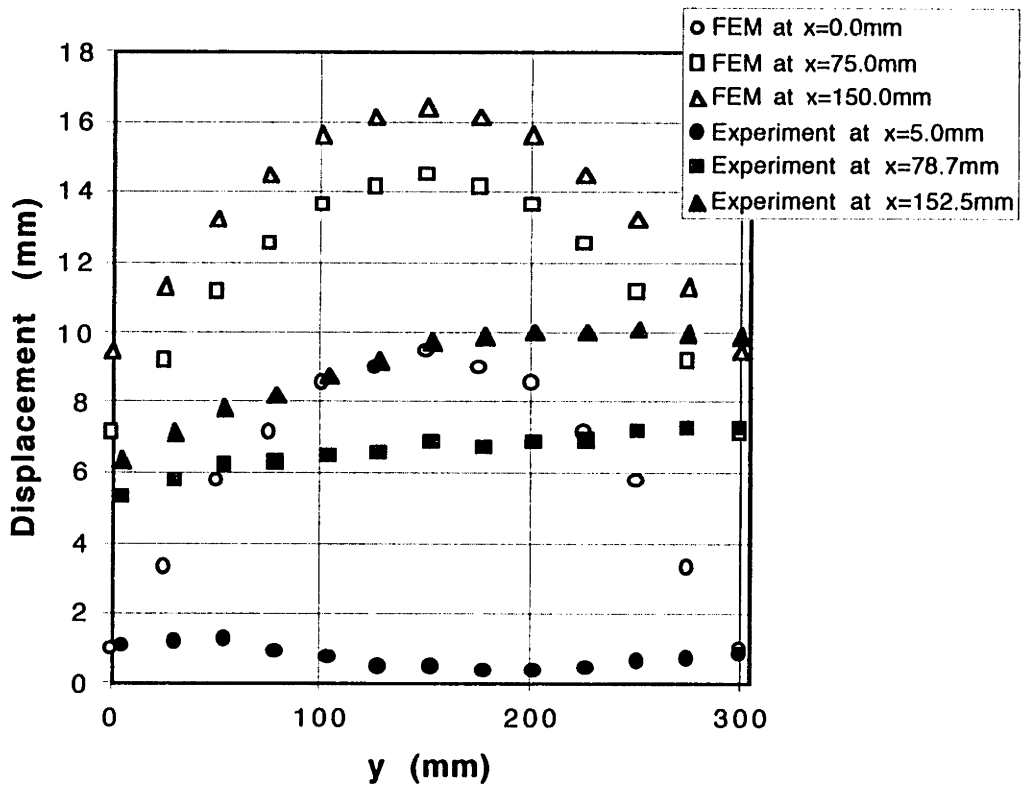


Figure 6-18: Comparison in profiles parallel to the y-axis between the FEM calculation result and the experimental result for multi heating lines Case 2

Multi Heating Line Case 3

The heating pattern for Case 3 is shown in Figure 6-19. The numbers next to the lines in the figure show the order of the processes and the arrows show the direction of the processes.

The calculation has been carried out under the following conditions:

- E=196000N/mm^2,
- ν=0.3,
- Dimensions=300mm $\times$ 300mm,
- Thickness=6.35mm,
- Number of Elements=144,
- Number of Nodes=481,
- Size of Elements=25.0 $\times$ 25.0mm,
- Bending Moment per Unit Length for Each Heating Line: 7591 N-mm/mm,
- Separation Length of Bending Moment (a) =12.5mm.

The FEM calculation deal with lines 1 and 2, lines 3 and 4, lines 6 and 5, lines 7 and 8, as continuous lines.

The experiment was conducted under the following conditions:

- Material: Inconel 625,
- Dimensions=304.8mm $\times$ 304.8mm,
- Thickness=6.35mm,
- Laser Beam: Nd:YAG Laser Beam,
- Output of a Laser Beam=2.5kW,
- Spot Size of a Laser Beam=20mm,

- Process Speed = 3.33mm/sec.

The shape of a plate calculated by the FEM program is shown in Figure 6-20. The experimental final shape processed by an Nd:YAG laser beam is shown in Figure 6-21.

Figure 6-22 shows the comparison of the profiles parallel to the x-axis. Figure 6-23 shows the comparison of the profiles parallel to the y-axis. The FEM result does not match the experimental result in this case.

When we see Figure 6-23 closely, the profiles of the experimental result at x=5.0mm and x=78.7 are concave shapes while those of the FEM calculation are convex shapes. This opposite direction of deformation seems to be caused by buckling of the plate during the heating processes.

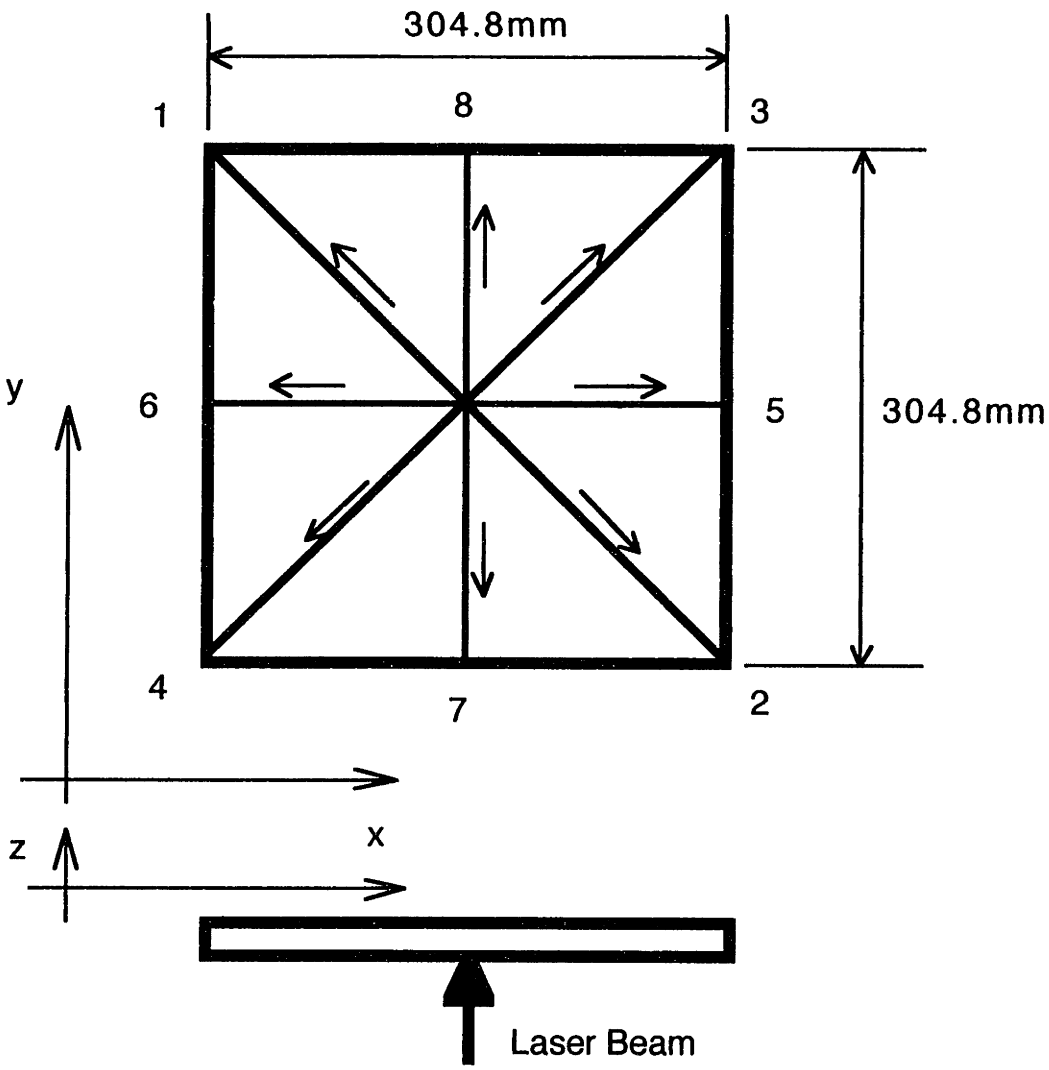


Figure 6-19: Heating pattern for multi heating lines Case 3 in Inconel 625

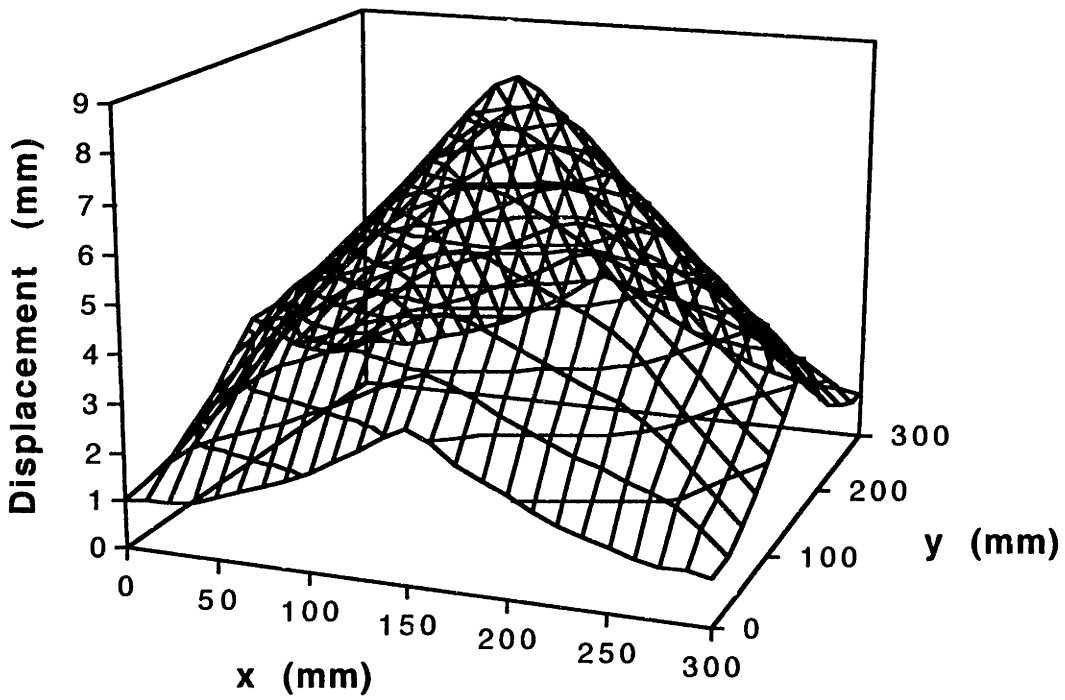


Figure 6-20: FEM calculation result for multi heating lines Case 3

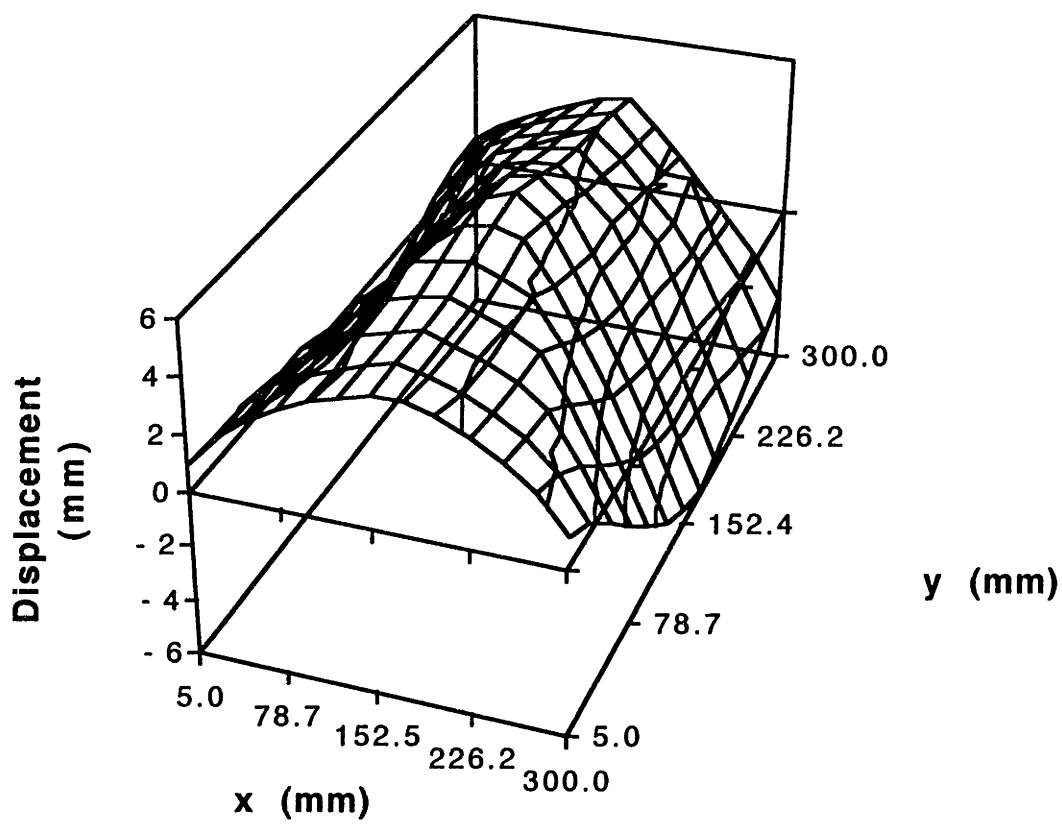


Figure 6-21: Experimental result for multi heating lines Case 3

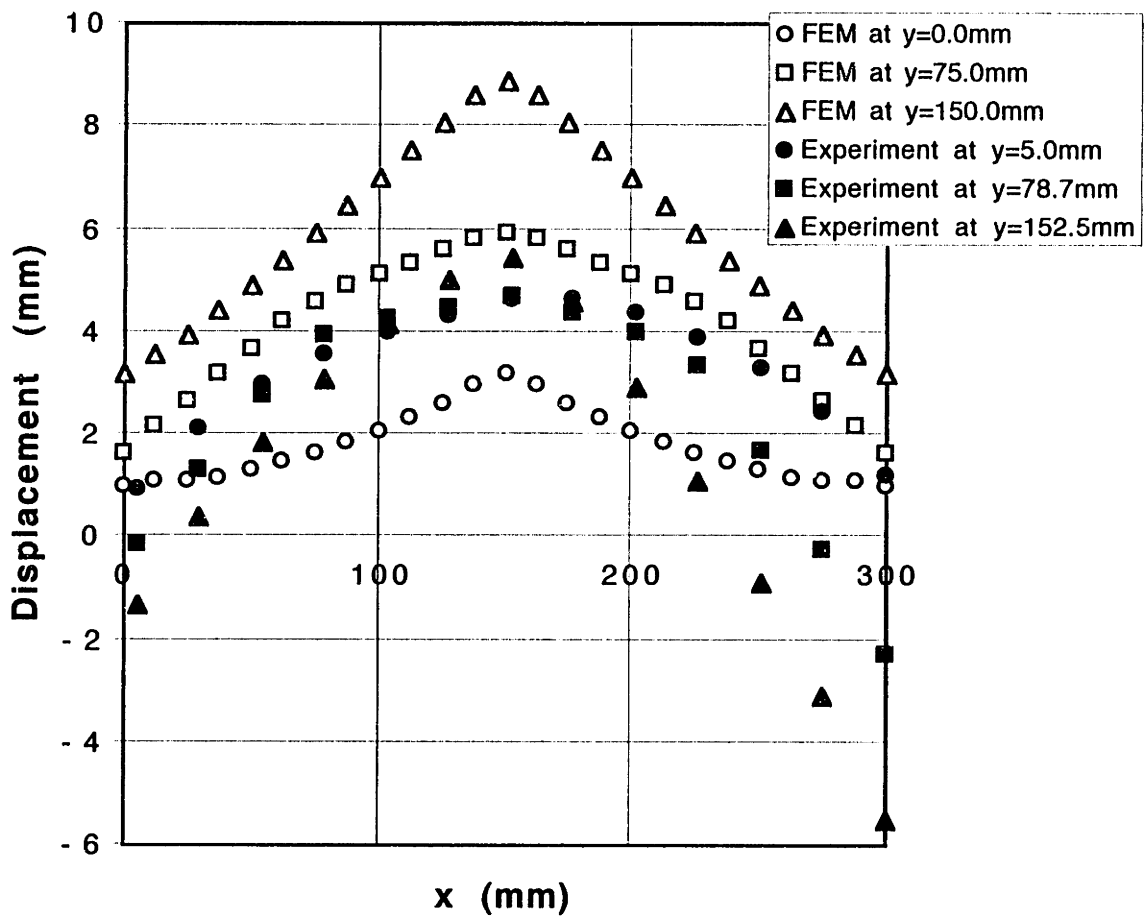


Figure 6-22: Comparison in profiles parallel to the x-axis between the FEM calculation result and the experimental result for multi heating lines Case 3

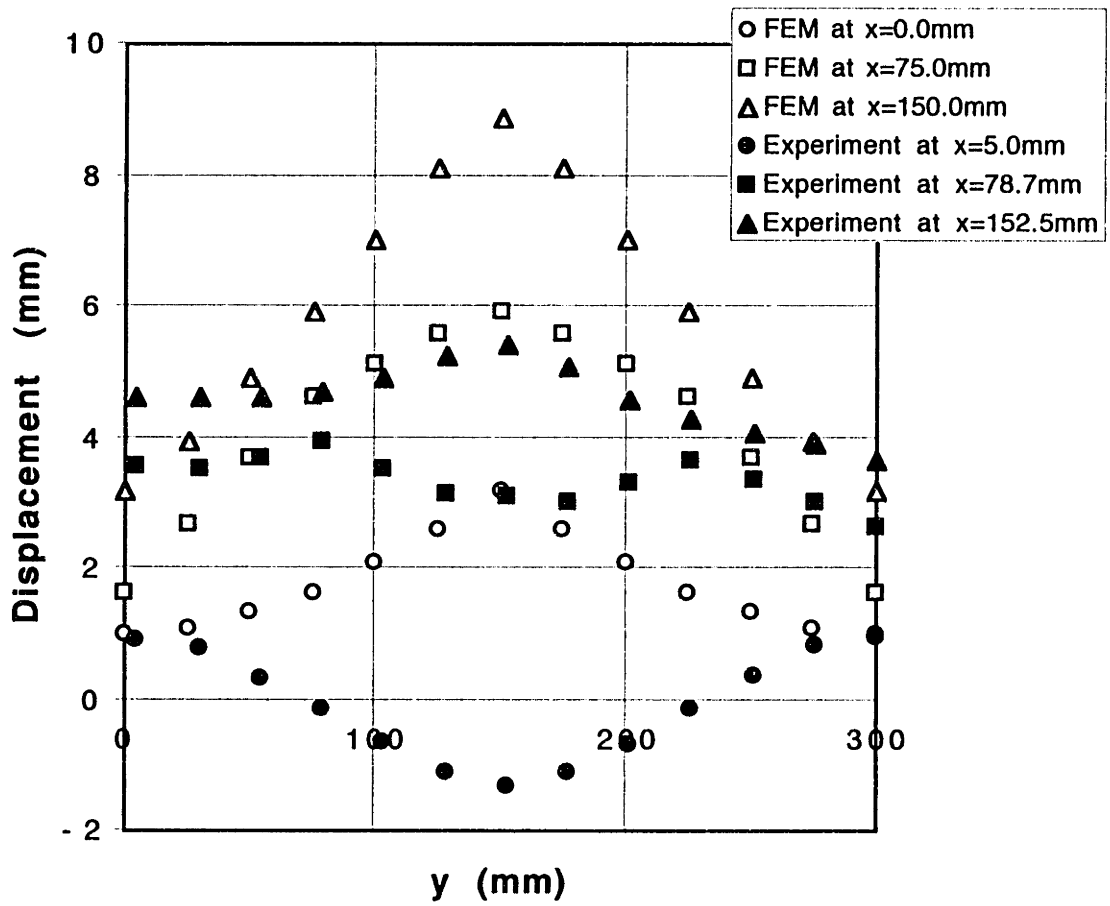


Figure 6-23: Comparison in profiles parallel to the y-axis between the FEM calculation result and the experimental result for multi heating lines Case 3

6.2 Validation of the Optimizing Program

The validation process consists of two parts: cases with using the results calculated by the FEM program as target shapes and a case with using an experimental result processed by a laser beam.

In the cases with the FEM results, a calculation result with known heating conditions was provided to the heat condition program as a target shape. Then, the program tried to determine the heating conditions that will shape a plate into the target shape. Since the decoding program in the heat condition program uses the FEM program that calculated the target shape, the expected shape obtained by the program as the solution of heating processes should exactly match the target shape. It means that if the program reaches the known answer of heating conditions, the fitness value should be equal to 1.

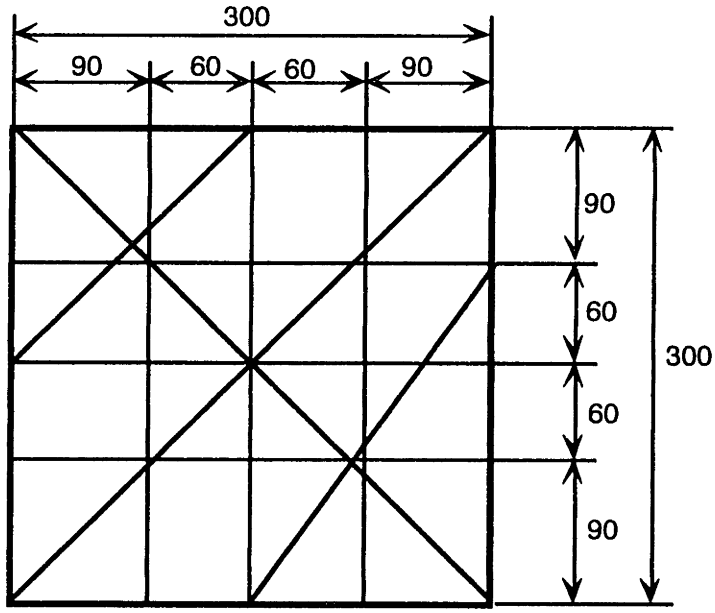
On the other hand, in the case with an experimental result, although the heat condition determination program tried to optimize the fitness value, the final fitness value does not reach 1 due to the error between the result calculated by the FEM program and the experimental result.

6.2.1 Cases with Shapes Obtained by the FEM Program as the Target Shapes

This section presents examples for the determination of heat input values for a given set of positions of heating lines. Ten heating lines are given as possible processing lines in these cases. Figure 6-24 shows these ten positions of heating lines. Each possible processing line has four possibilities: processing two times from the upper side of a plate, processing one time from the upper side, no processing, and processing one time from the lower side. The total number of combinations of heating processes is 4^{10}= 1,048,576.

Three shapes that had been obtained as results of calculation using the FEM subprogram were given as target shapes;

- Case 1: Two heating lines at positions 7 and 8,



Line Position	Point 1	Point 2
1	(90,0)	(90, 300)
2	(150,0)	(150,300)
3	(210,0)	(210,300)
4	(0,90)	(300,90)
5	(0,150)	(300,150)
6	(0,210)	(300,210)
7	(0,0)	(300,300)
8	(0,300)	(300,0)
9	(0,150)	(150,300)
10	(150,0)	(300,210)

Possibilities for Each Heating Line:

"00" means putting a heat process on the reverse side,
"01" means no heat process,
"10" means putting a heat process on the front side,
"11" means putting two heat processes on the front side.

Figure 6-24: Heating line positions for validation of the optimizing program

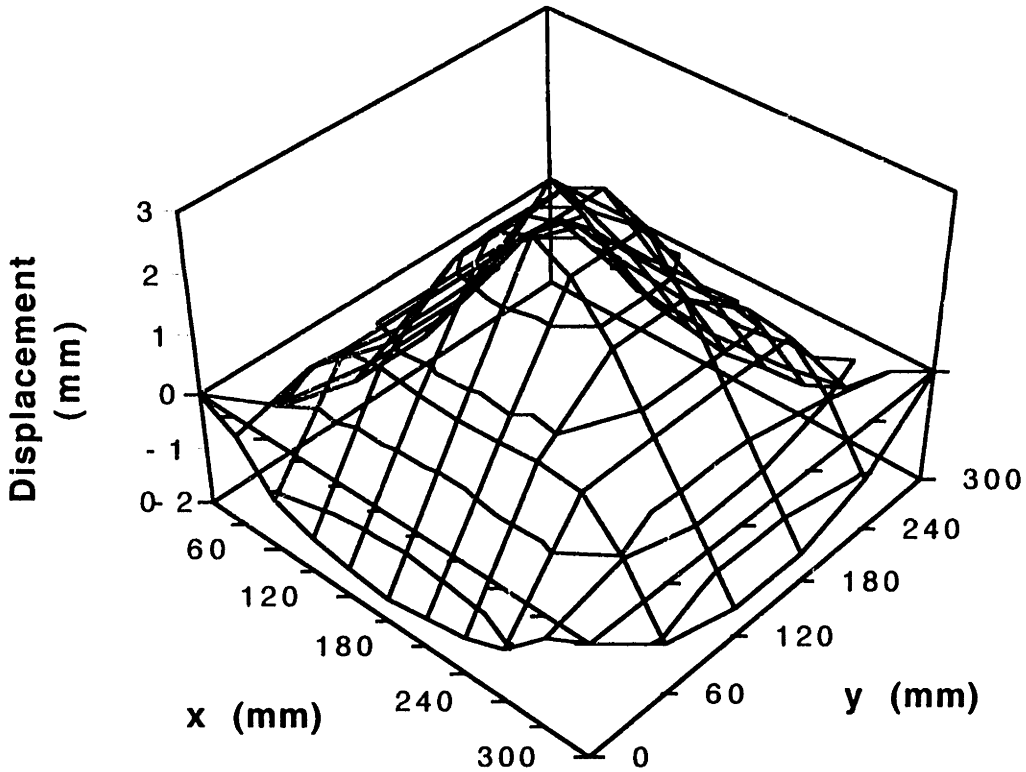


Figure 6-25: Target shape of Case 1

- Case 2: Six heating lines at positions 2, 5, 7, 8, 9, and 10,
- Case 3: Ten heating lines at positions from 1 through 10.

The number of nodes is 96 and the number of elements is 25 for each case. Figure 6-25, 6-26, and 6-27 show the target shapes of the cases. These shapes were obtained by the liner FEM subprogram developed in the previous progress report.

As shown in Section 5.2.2, the fitness value, FV, is expressed as follows:

$$FV = \frac{1}{1 + |TD|/(t \cdot N)}, \tag{6.2}$$

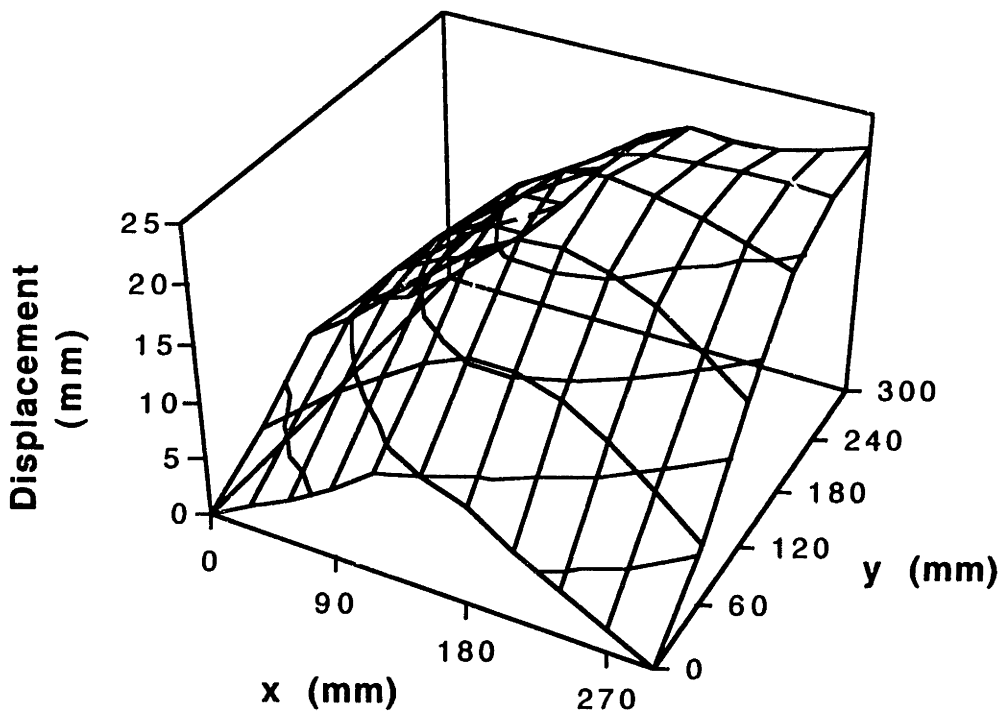


Figure 6-26: Target shape of Case 2

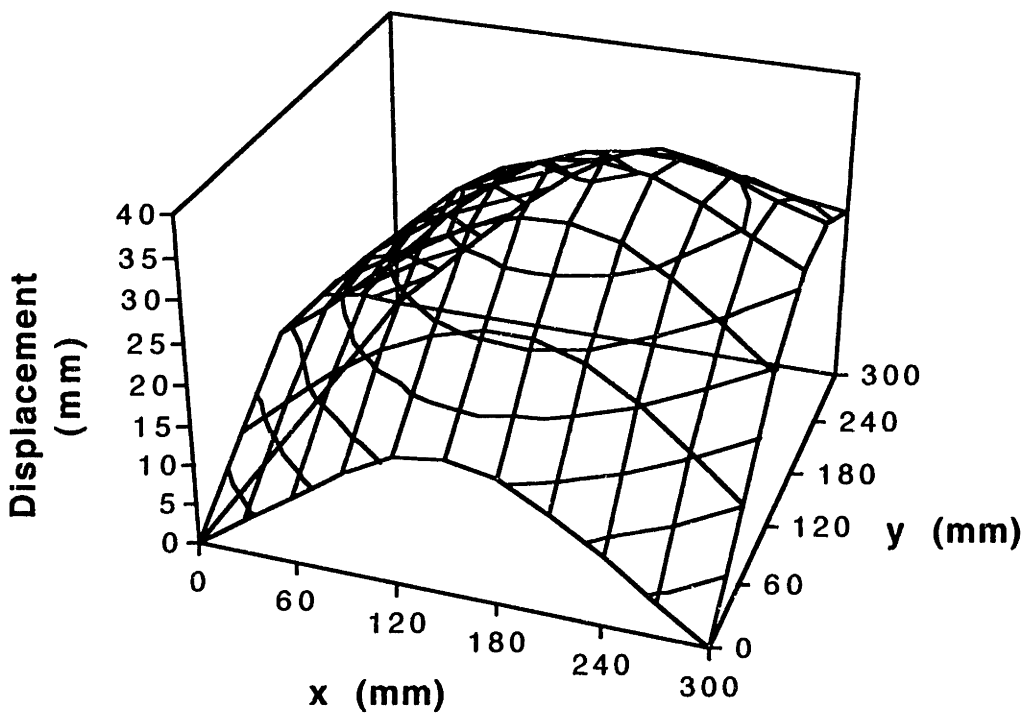


Figure 6-27: Target shape of Case 3

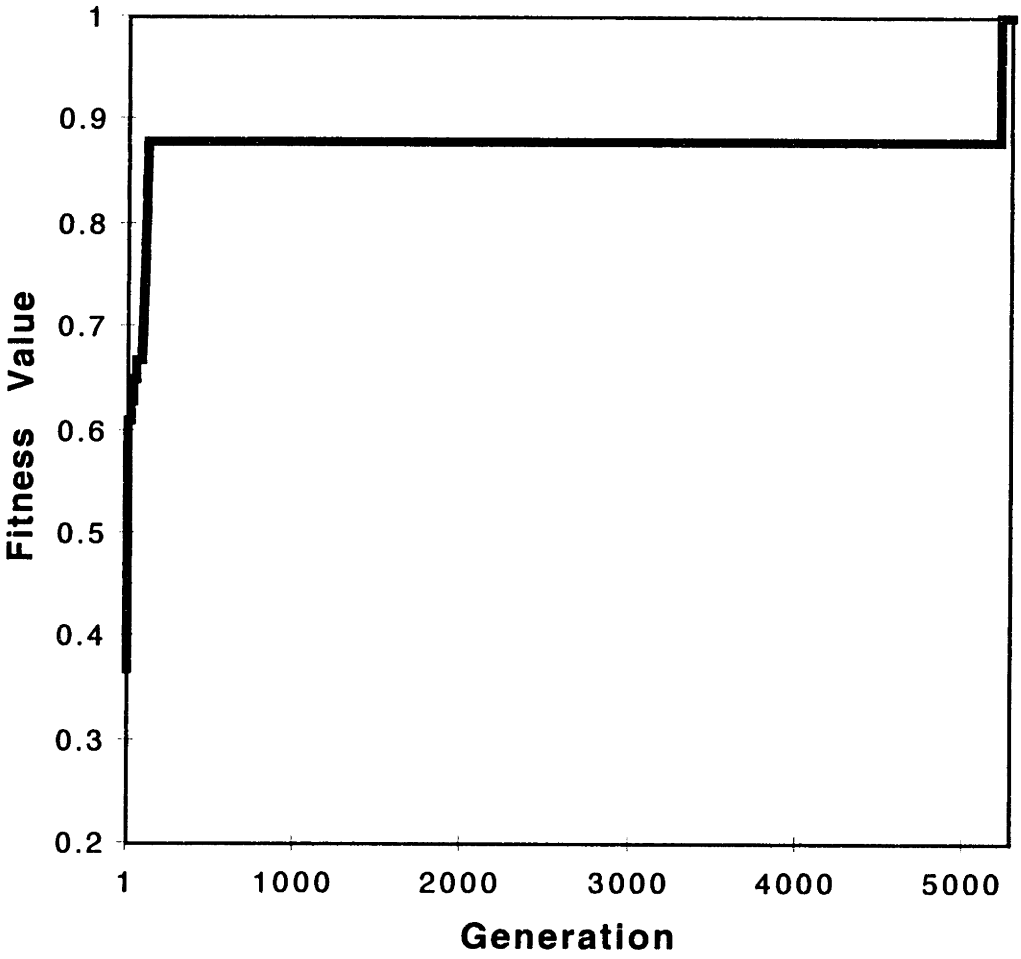


Figure 6-28: Change of fitness value over generations in Case 1

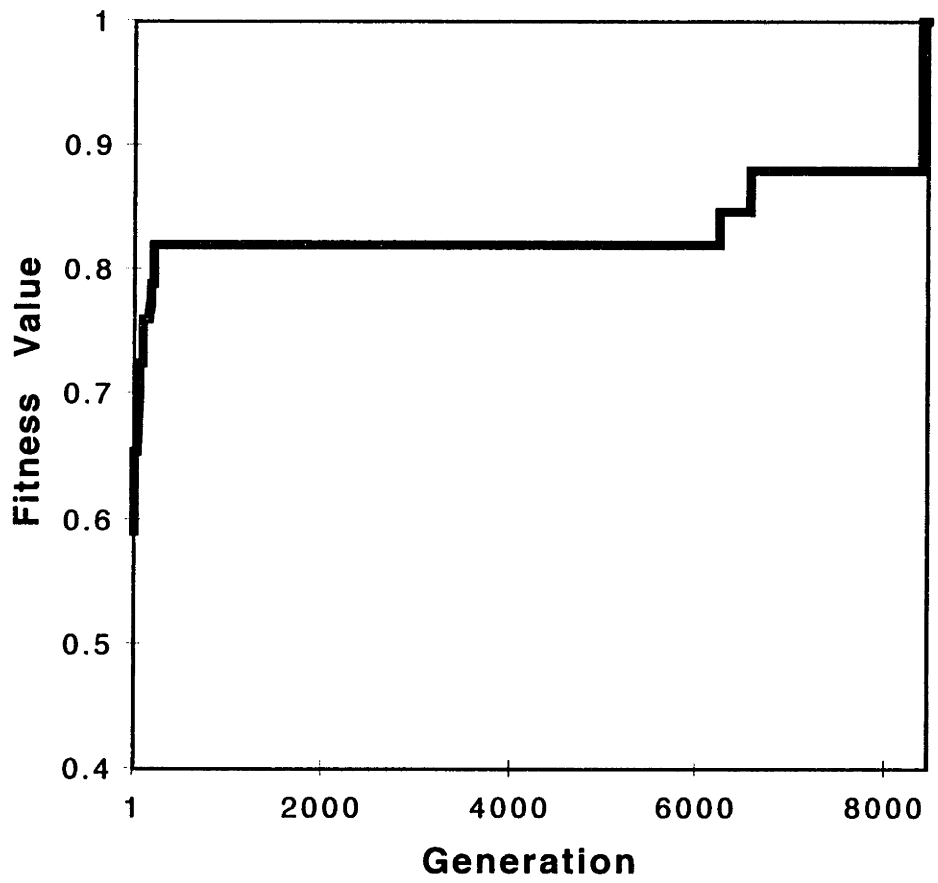


Figure 6-29: Change of fitness value over generations in Case 2

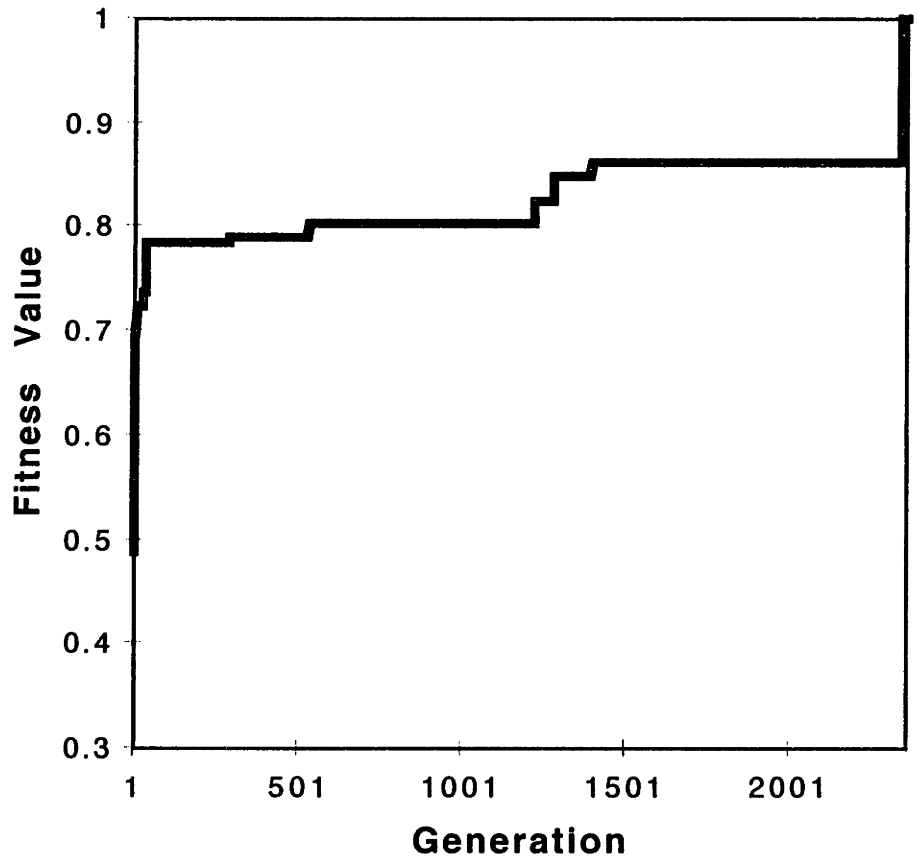


Figure 6-30: Change of fitness value over generations in Case 3

where TD is the sum of absolute values of the differences between an expected final shape and the target shape, t is thickness of the plate, and N is the number of reference points that is used to obtain TD. If an expected final shape matches the target shape completely, TD is equal to zero. Then the value of FV is 1. Therefore, we can know that the program reaches the correct answer when the FV becomes 1.

The change of fitness value in case 1 is shown in Figure 6-28. The value of FV increases intermittently with the generation. In order to solve case 1, the program needed 5,523 generations. Each generation has the population of 5. This means that 27,615 ($= 5523 \times 5$) calculations were needed to solve the case. Out of 1,048,576 ($= 4^{10}$) possibilities, the program obtained the correct answer with the search of 2.63 % ($= 27615/1048576 \times 100$) area in the total search space.

The change of fitness value in case 2 is shown in Figure 6-29 and Figure 6-30 shows the change in case 3. Case 2 and 3 were solved at the generation of 8,367 and 2,345 respectively with the population of 5 for each generation. These calculations are 3.99 % and 1.12 % of the total search space. Although the total calculation times were not measured, they were roughly in the rage of from 3 to 6 hours on an HP 9000/715 workstation.

These results show that the GA program could properly determine the heat process conditions for all the target shapes.

6.2.2 Case with an Experimental Final Shape as the Target Shape

In this case, the result presented in Section 6.1.3 was used as a target shape. The heating pattern is shown in Figure 6-32. The numbers next to the lines in the figure show the order of the processes and the arrows show the direction of the processes. Lines 7 to 10 were added to the heating pattern as dummy heating lines even though those were not utilized in the experiment.

The calculation was carried out under the following conditions:

- E=196000N/mm^2,
- ν=0.3,
- Dimensions=300mm $\times$ 300mm,
- Thickness=6.35mm,
- Number of Elements=144,
- Number of Nodes=481,
- Size of Elements=25.0 $\times$ 25.0mm,
- Bending Moment per Unit Length for 100% intensity: 7951 N-mm/mm,
- Separation Length of Bending Moment (a) =12.5mm.
- Possibilities for Each Heating Line: 100%, 80%, 60%, 40%, 20%, 0%, -20%, and -40% (total 8 possibilities).

The experiment was conducted under the following conditions:

- Material: Inconel 625,
- Dimensions=304.8mm $\times$ 304.8mm,
- Thickness=6.35mm,

- Laser Beam: Nd:YAG Laser Beam,
- Output of a Laser Beam=2.5kW,
- Spot Size of a Laser Beam=20mm,
- Process Speed for Line 1, 5, and 6: 5.0mm/sec (80%),
- Process Speed for Line 2, 3, and 4: 3.33mm/sec (100%).

Since the laser beam was traversed at two different speeds on the plate, the bending moment per unit length has two values. The ratio of the values of the bending moment per unit length was determined by the following assumption:

$$MPU = A\frac{P}{t\sqrt{v}} \tag{6.3}$$

where MPU is the total bending moment per unit length, P is the power of the laser beam, t is the thickness of the plate, v is the process speed, all in SI units, and A is a constant that depends upon the heat intensity and the spot size of the laser beam, and the absorptivity of the plate.

The above assumption is based on the relationship between angular distortion and process parameter, $P/(t\sqrt{v})$, obtained by Masubuchi et al [27]. Although constant value, A, should be determined by series of experiments, it was assumed to be equal to 1 $mm^{0.5}s^{0.5}$ in the following calculation.

The change of fitness value up to 10,000 generations is shown in Figure 6-31. The total calculation time for 10,000 generations was approximately 50 hours on an HP 9000/715 workstation. The shape of a plate obtained by the heat condition determination program after 10,000 generations is shown in Figure 6-33. The experimental final shape processed by an Nd:YAG laser beam is shown in Figure 6-11. The heat condition answer obtained by the program were shown with the experimental heat conditions in parentheses as follows:

- Heating Line 1: 60% (80%)

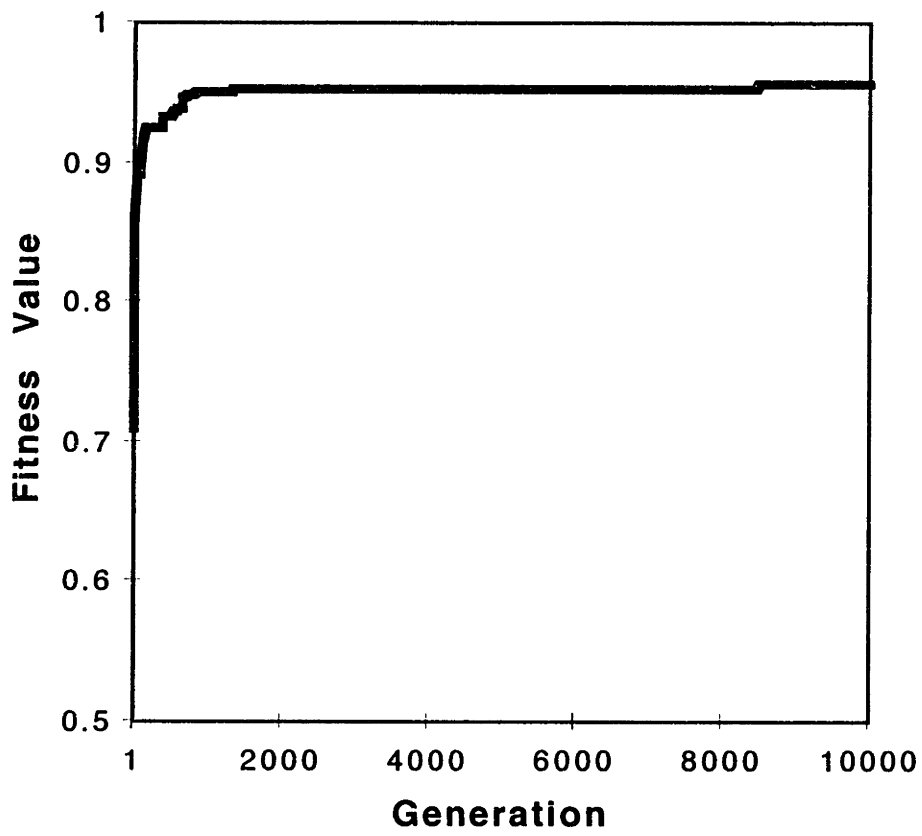


Figure 6-31: Change of fitness value over generations up to 10,000 generations

- Heating Line 2: 40% (100%)
- Heating Line 3: 40% (100%)
- Heating Line 4: 0% (100%)
- Heating Line 5: 60% (80%)
- Heating Line 6: 40% (80%)
- Heating Line 7: 40% (0%)
- Heating Line 8: 60% (0%)
- Heating Line 9: 80% (0%)
- Heating Line 10: 80% (0%)

The heating conditions obtained by the program after 10,000 generations are quite different from the actual heating conditions in the experiment. However, if we closely examine the expected shape under the above conditions calculated by the decoding program, we realize that the shape is close to the target shape.

Figure 6-34 shows the comparison of the profiles parallel to the x-axis. Figure 6-35 shows the comparison of the profiles parallel to the y-axis. The comparison in Figure 6-35 shows good agreement between the two shapes within 0.5 mm out of 7 mm at the center; and 0.5 mm out of 4 mm at the edge. Although the comparison in Figure 6-34 shows the difference at the edge of the plate, the profile at the center calculated for the resultant heat processes matches that of the target shape within 0.5 mm out of 7 mm. It shows that the program optimized heating conditions properly, although the resultant heating conditions optimized by the program are different from the actual conditions.

Obviously, the different heating conditions from the actual conditions have been obtained due to the calculation error of the decoding program and the assumption expressed by Equation 6.3. If we improve the accuracy of the decoding program and obtain the precise relationship between bending moment and process parameters by

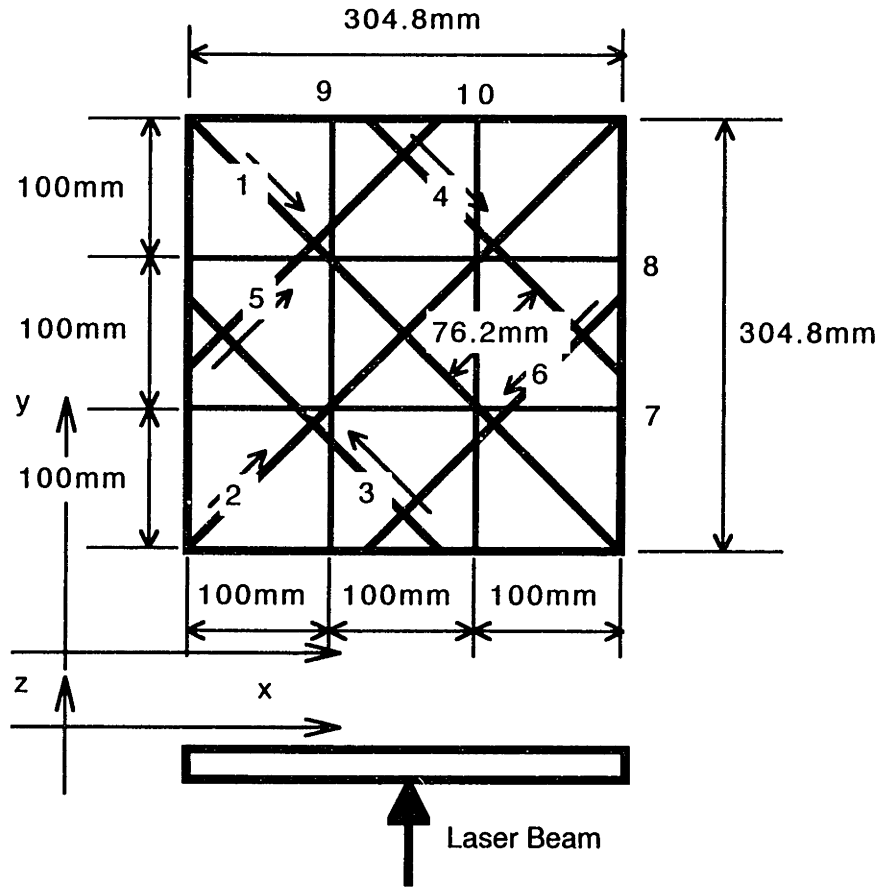


Figure 6-32: Heating pattern for multi heating lines

experiments, the result obtained by the program should match the actual heating conditions.

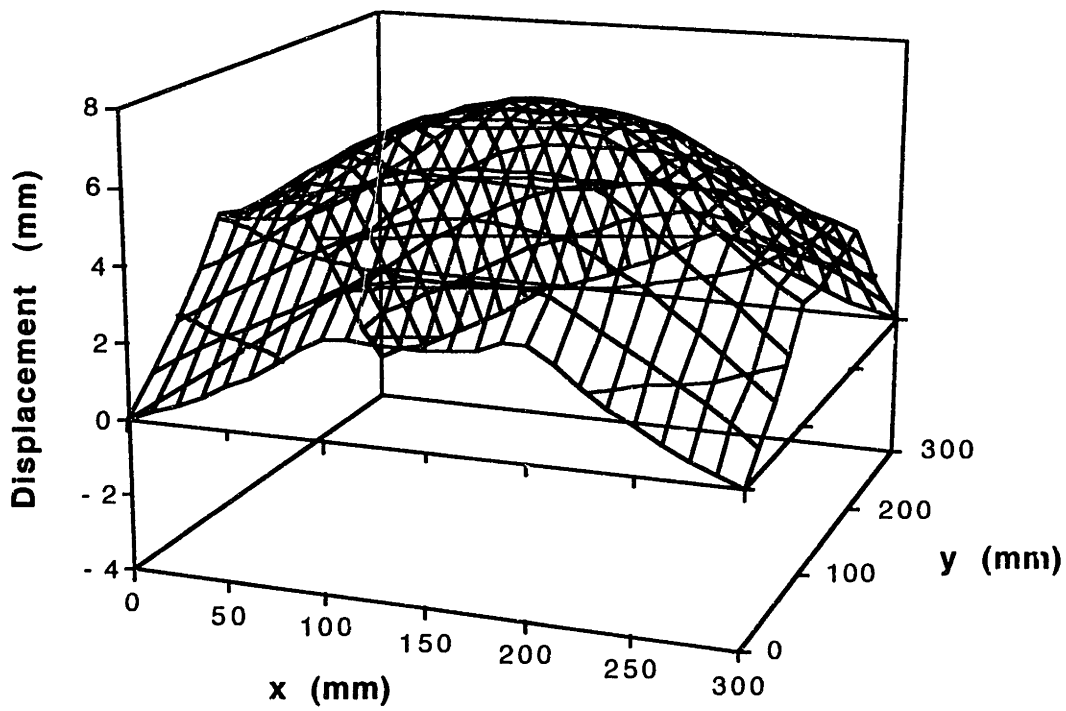


Figure 6-33: Final shape obtained by GA after 10,000 generations

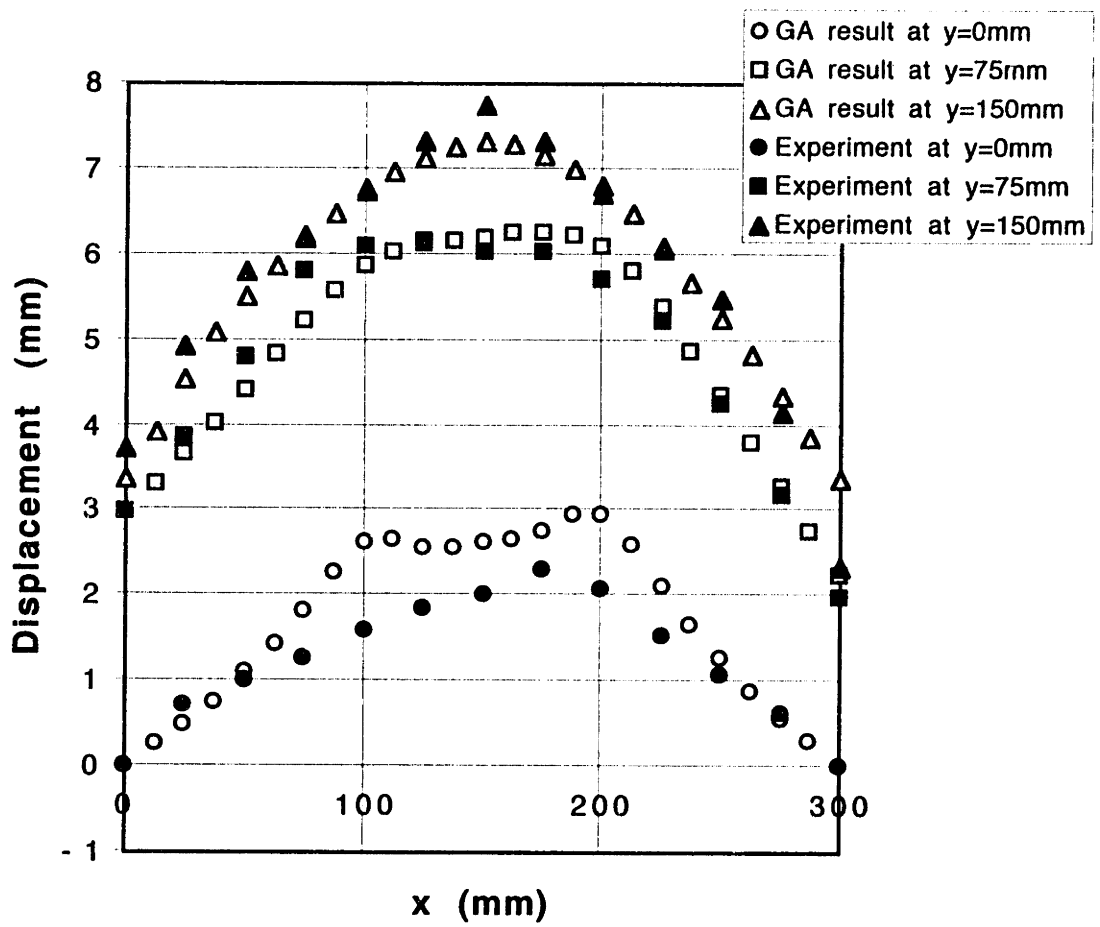


Figure 6-34: Comparison in profiles parallel to the x-axis between the result by GA and the experimental result

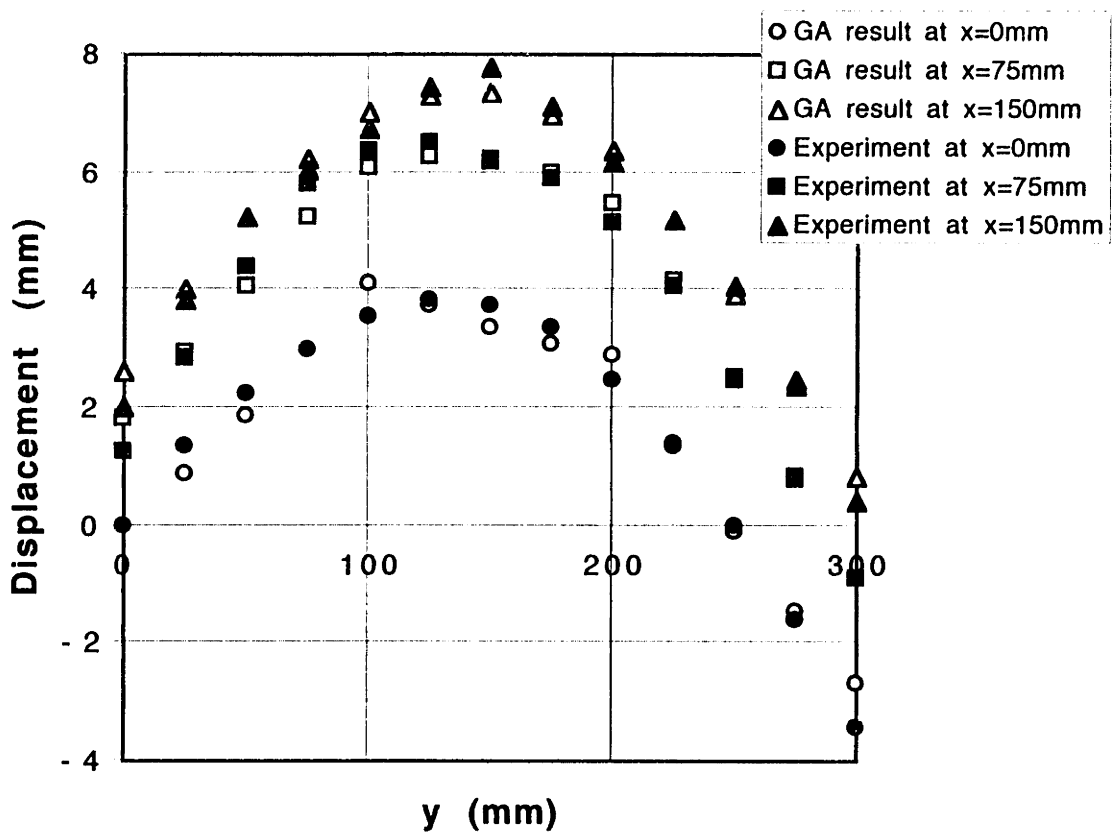


Figure 6-35: Comparison in profiles parallel to the y-axis between the result by GA and the experimental result

6.3 Summary

The validation of the linear FEM program, the decoding part of the process determination program, shows that the calculated shape does not match the experimental shape in the case that the effects of initial shape and buckling are critical, since the linear FEM decoding program cannot deal with those effects. However, the heating process determination program could obtain the correct set of heating lines for the target shape calculated by the linear FEM program. It needed several thousand generations to get the correct answer for a case with 5 populations and 10 heating lines of 4 possibilities. It could obtain the answers after the search of less than 4% area of the total search space. Also, the program tried to obtain the actual heat processes applied to a plate with an experimental shape as the target shape. Even though it failed to obtain the correct set of heating conditions due to the error of the decoding part and the assumed relationship between bending moment and process parameters, the optimizing part worked properly to solve the problem.

Chapter 7

Conclusion and Future Work

7.1 Conclusion

A new algorithm to determine a set of heat process conditions for forming by a moving heat source has been invented, implemented, and validated. The algorithm can optimize the heat input conditions for a given set of heating lines in order to obtain a desired shape, if the plate is nearly flat during the process, for instance, within 10mm in a 300mm square plate 6mm thick.

The new algorithm consists of an optimizing program that searches for better combination of heat process conditions randomly and a decoding program that transforms a string representing a set of heat process conditions into an expected final shape.

The algorithm has much flexibility to be implemented. It can be used to determine a set of heat process conditions for an arbitrary three dimensional shape.

The accuracy of the algorithm depends on that of a decoding program. If high accuracy is needed, a decoding program with high accuracy has to be used.

The algorithm has been implemented with a combination of a simple linear, inextensible plate FEM program and an optimizing program based on a Genetic Algorithm.

Final deformation due to a heating line is simulated by applying a pair of bending moment rows facing each other in a linear FEM program. This enables us to use

a GA to determine heating process conditions by reducing the computation time significantly.

Its usefulness has been demonstrated by comparison with the calculation results and the actual final shapes, subject to the condition that the initial shape effect on bending is small and there is no buckling of a plate.

7.2 Future Work

The program has been developed in a complete manner. Still, it has several things to be implemented for practical uses.

7.2.1 Improvement of a Decoding Program

The error between the calculation and the experimental result is large when the initial shape effect on the stiffness of a plate and the buckling of the plate are critical, since it uses a highly simplified model in order to keep a calculation time reasonably short. We need to improve the accuracy of the calculation by considering the order of heating lines and using shell elements, that can simulate the buckling of a plate, instead of plate elements. This improvement is relatively easy because the decoding program can be replaced once we obtain a better FEM program.

7.2.2 Selection of the Fitness Function and Convergence Criterion

At this time, the program does not have a criterion to finish the calculation. We can recognize the finish of the calculation only when we see the correct set of heat processes in the output file. If we do not know the correct answer which should have maximum fitness value for the target shape, the program does not stop the calculation.

In practical uses, as an operator does not know the combination of heat processes that produces the maximum fitness value, the program needs a certain criterion as-

sociated with the designed accuracy to stop calculations. Although the above fitness function worked for the target shapes obtained by FEM calculations, considering the practical uses, in order to achieve the desired accuracy, we need to have a quantitative fitness value. This enables us to finish the calculation when the program obtains the heat process set that produces the shape with satisfactory accuracy.

7.2.3 Partial Heating Lines

The program has the limitation on the positions of heating lines; the end points of a heating line have to be located on the edges of a plate. The program cannot deal with a spoke-like pattern of heat processes, since the heat lines stop in the middle of a plate. It is necessary to add the function that can deal with partial heating lines into the program for broader use of the program.

7.2.4 Optimization of GA Parameters

The number of generations to reach the right answer is related to the population of strings for a generation, the mutation rate, the type of crossover, etc. Therefore, it is necessary to optimize these values to obtain the right results for all cases with as few generations as possible. There is possibility that the calculation time would be reduced by optimization of GA parameters combining the selection of fitness function.

Appendix A

Process Determination Program

A.1 Input Variable Definitions Required by Subroutine "GA_input"

irestrt = 0 for a new GA run, or for a single function evaluation ; = 1 for a restart continuation of a GA run.

npopsiz The population size of a GA run (typically 100 works well). For a single calculation, set equal to 1.

nparam Number of parameters (groups of bits) of each individual. Make sure that nparam matches the number of values in the parmin, parmax and nposibl input arrays.

maxgen The maximum number of generations to run by the GA. For a single function evaluation, set equal to 1.

idum The initial random number seed for the GA run. Must equal a negative integer, e.g. idum=-1000.

pmutate The jump mutation probability. Typically set = 1/npopsiz.

pcreep The creep mutation probability. Typically set this = (nchrome/nparam)/npopsiz.

pcross The crossover probability. For single-point crossover, a value of 0.6 or 0.7 is recommended. For uniform crossover, a value of 0.5 is suggested.

itourny No longer used. The GA is presently set up for only tournament selection.

ielite = 0 for no elitism (best individual not necessarily replicated from one generation to the next). ; = 1 for elitism to be invoked (best individual replicated into next generation); elitism is recommended.

iunifrm = 0 for single-point crossover ; = 1 for uniform crossover; uniform crossover is recommended.

iniche = 0 for no niching ; = 1 for niching; niching is recommended.

nchild = 1 for one child per pair of parents (this is what I typically use). ; = 2 for two children per pair of parents (2 is more common in GA work).

iskip = 0 for normal GA run (this is standard). ; = number in population to look at a specific individual or set of individuals. Setting iskip-0 is only used for debugging purposes.

iend = 0 for normal GA run (this is standard). ; = number of last population member to be looked at in a set of individuals. Setting iend-0 is only used for debugging purposes and is commonly used in conjunction with iskip.

nowrite = 0 to write detailed mutation and parameter adjustments ; = 1 to not write detailed mutation and parameter adjustments

parmin = array of the minimum allowed values of the parameters

parmax = array of the maximum allowed values of the parameters

nposibl = array of integer number of possibilities per parameter. For optimal code efficiency set nposibl=2^n, i.e. 2, 4, 8, 16, 32, 64, etc.

nichflg = array of 1/0 flags for whether or not niching occurs on a particular parameter. Set to 0 for no niching on a parameter, set to 1 for niching to operate

on parameter. The default value is 1, but the implementation of niching is still controlled by the flag iniche.

microga = 0 for normal conventional GA operation ; = 1 for micro-GA operation (this will automatically reset some of the other input flags). I recommend using npopsiz=5 when microga=1.

A.2 Input Variable Definitions Required by Subroutine "INPUT"

NPROB Total number of problems to be solved in one run. No longer used.

TITLE Title of the problem - limited to 72 alphanumeric characters.

NPOIN Total number of nodal points.

NELEM Total number of elements.

NVFIX Total number of restrained boundary points where one or more degrees of freedom are restrained.

NCASE Total number of load cases to be analyzed. No longer used.

NTYPE Problem type parameter.

NNODE Number of nodes per element (=8).

NDOFN Number of degrees of freedom per node (=2).

NMATS Total number of different materials.

NPROP Number of independent properties per material (=4).

NGAUS Order of integration formula for numerical integration.

NDIME Number of coordinate dimensions (=2).

NSTRE Number of independent stress components (=3).

NUMEL Element number.

MATNO(NUMEL) Material property number.

LNODS(NUMEL,1) 1st Nodal connection number.

LNODS(NUMEL,2) 2nd Nodal connection number.

⋮

LNODS(NUMEL,8) 8th Nodal connection number.

IPOIN Nodal point number.

COORD(IPOIN,1) x-coordinate of node.

COORD(IPOIN,2) y-coordinate of node.

NOFIX(IVFIX) Restrained node number.

IFPRE(IVFIX,1) Condition of restraint on nodal displacement, w. =0 No displacement restraint; =1 Nodal displacement restrained.

IFPRE(IVFIX,2) Condition of restraint on nodal rotation, θ_x. =0 No displacement restraint; =1 Nodal displacement restrained.

IFPRE(IVFIX,3) Condition of restraint on nodal rotation, θ_y. =0 No displacement restraint; =1 Nodal displacement restrained.

PRESC(IVFIX,1) The prescribed value of nodal displacement, w.

PRESC(IVFIX,2) The prescribed value of nodal rotation, θ_x.

PRESC(IVFIX,3) The prescribed value of nodal rotation, θ_y.

NUMAT Material identification number.

PROPS(NUMAT,1) Elastic modulus, E.

PROPS(NUMAT,2) Poisson's ratio, ν.

PROPS(NUMAT,3) Material thickness, t.

PROPS(NUMAT,4) Intensity of any uniformly distributed load. No longer used.

DISTA Separate distance for bending moment.

SW Allowable distance for allying bending moment.

THATDEG Angular distortion for a standard heat process in degree.

STANDWIDTH Plate width of a plate for a standard heat process in mm.

ITNL Total number of heating lines.

LN Line number for heating lines.

X_1 x-coordinate of an edge point of a heating line.

Y_1 y-coordinate of an edge point of a heating line.

X_2 x-coordinate of another edge point of the heating line.

Y_2 y-coordinate of another edge point of the heating line.

ii Node number for a target shape.

TARSHAPE(ii) Displacement of a target shape for node ii.

A.3 Code variable Definitions for the GA Part

best = the best fitness of the generation

child = the floating point parameter array of the children

cpu = cpu time of the calculation

creep = +1 or -1, indicates which direction parameter creeps

del = square root of del2

del2 = sum of the squares of the normalized multidimensional distance between member j and all other members of the population

delta = del/nparam

diffrac = fraction of total number of bits which are different between the best and the rest of the micro-GA population. Population convergence arbitrarily set as diffrac < 0.05.

fbar = average fitness of population

fitness = array of fitnesses of the parents

fitsum = sum of the fitnesses of the parents

g0 = lower bound values of the parameter array to be optimized. The number of parameters in the array should match the dimension set in the above parameter statement.

g1 = the increment by which the parameter array is increased from the lower bound values in the g0 array. The minimum parameter value is g0 and the maximum parameter value equals g0+g1 $\times(2^{g2} - 1)$, i.e. g1 is the incremental value between min and max.

ig2 = array of the number of bits per parameter, i.e. the number of possible values per parameter. For example, ig2=2 is equivalent to 4 ($=2^2$) possibilities, ig2=4 is equivalent to 16 ($=2^4$) possibilities.

ig2sum = sum of the number of possibilities of ig2 array

ibest = binary array of chromosomes of the best individual

ichild = binary array of chromosomes of the children

icount = counter of number of different bits between best individual and other members of micro-GA population

icross = the crossover point in single-point crossover

indmax = maximum number of individuals allowed, i.e. max population size

iparent = binary array of chromosomes of the parents

istart = the generation to be started from

jbest = the member in the population with the best fitness

jelite = a counter which tracks the number of bits of an individual which match those of the best individual

jend = used in conjunction with iend for debugging

jstart = used in conjunction with iskip for debugging

kount = a counter which controls how frequently the restart file is written

kountmx = the maximum value of kount before a new restart file is written; presently set to write every fifth generation. Increasing this value will reduce I/O time requirements and reduce wear and tear on your storage device

kelite = kelite set to unity when jelite=nchrome, indicates that the best parent was replicated amongst the children

mate1 = the number of the population member chosen as mate1

mate2 = the number of the population member chosen as mate2

nchrmax = maximum number of chromosomes (binary bits) per individual

nchrome = number of chromosomes (binary bits) of each individual

ncreep = number of creep mutations which occurred during reproduction

nmutate = number of jump mutations which occurred during reproduction

nparmax = maximum number of parameters which the chromosomes make up

paramav = the average of each parameter in the population

paramsm = the sum of each parameter in the population

parent = the floating point parameter array of the parents

pardel = array of the difference between parmax and parmin

rand = the value of the current random number

sigshar = floating point equivalent of nparam

sumshar = the scaling factor to be applied to the fitness of each individual based on a triangular sharing function

npossum = sum of the number of possible values of all parameters

time0 = clock time at start of run

A.4 Explanations of Subroutines of the GA Part

code codes floating point value to binary string.

crosovr performs crossover (single-point or uniform).

decode decodes binary string to floating point value.

evalout evaluates the fitness of each individual and outputs generational information to the 'ga.out' file.

func The function which is being evaluated.

gamicro implements the micro-GA technique.

input inputs information from the 'ga.inp' file.

initial Program initialization and inputs information from the 'ga.restart' file.

mutate performs mutation (jump and/or creep).

newgen writes child array back into parent array for new generation; also checks to see if best individual was replicated (elitism).

niche performs niching (sharing) on population.

possibl checks to see if decoded binary string falls within specified range of parmin and parmax.

ran3 The random number generator.

restart writes the 'ga.restart' file.

select A subroutine of 'selectn'.

selectn performs selection; tournament selection is the only option in this version of the code.

shuffle shuffles the population randomly for selection.

A.5 Explanations of Subroutines of the FEM Decoding Program

INPUT inputs the control data, geometric data, boundary conditions, material properties, and information needed to calculate the total bending moment for one heating line.

NODEXY generates the coordinates of midside nodes which lie on a straight line connecting adjacent corner nodes.

GAUSSQ generates the sampling point positions and weighting factors according to the order of integration rule specified in the control data.

BMATPB calculates the strain matrix B for the plate bending element using the shape functions and their Cartesian derivatives.

MODPB(LPROP) calculates the coefficients of the matrix of elastic rigidities D for the current element using the element material properties.

STIFPB calculates the element stiffness and stress matrices.

SFR2(S,T) computes the shape functions and their local derivatives.

JACOB2(IELEM,DJACB,KGASP) calculates the coordinates of the Gauss points, the Jacobian matrix, the inverse of the Jacobian matrix, and the Cartesian shape function derivatives.

DBE computes the matrix product DB required for stress determination.

STREPB calculates the stress resultants at the Gauss points.

LOADPB(I) evaluates equivalent nodal forces and input data for discrete nodal loading.

FRONT assembles the equations and eliminates the variables by the frontal solution method.

CHECK1 detects errors.

ECHO echoes the remainder of the problem input.

CHECK2 detects errors.

A.6 Source Code of the Program

```
c#######################################################################
c
      program HeatDesign
c
c     Written by Hideki Shimizu
c
c#######################################################################
c
      implicit double precision (a-h,o-z)
      save
c
      include 'params.f'
      dimension parent(indmax,nparmax),child(indmax,nparmax)
```

```
      dimension fitness(indmax),nposibl(nparmax),nichflg(nparmax)
      dimension iparent(indmax,nchrmax),ichild(indmax,nchrmax)
      dimension g0(nparmax),g1(nparmax),ig2(nparmax)
c      dimension paramsm(nparmax),paramav(nparmax),ibest(nchrmax)
      dimension ibest(nchrmax)
      dimension parmax(nparmax),parmin(nparmax),pardel(nparmax)
c
      DIMENSION TITLE(18)
c
      common / ga1   / npopsiz,nowrite
      common / ga2   / nparam,nchrome
      common / ga3   / parent,iparent
      common / ga4   / fitness
      common / ga5   / g0,g1,ig2
      common / ga6   / parmax,parmin,pardel,nposibl
      common / ga7   / child,ichild
      common / ga8   / nichflg
      common /inputga/ pcross,pmutate,pcreep,maxgen,idum,irestrt,
     +                 itourny,ielite,icreep,iunifrm,iniche,
     +                 iskip,iend,nchild,microga,kountmx
c
      COMMON/CONTRO/NPOIN,NELEM,NNODE,NDOFN,NDIME,
     +     NSTRE,NTYPE,NGAUS,NPROP,NMATS,NVFIX,NEVAB,
     +     ICASE,NCASE,ITEMP,IPROB,NPROB
      COMMON/LGDATA/COORD(200,2),PROPS(10,
     +     4),PRESC(40,3),ASDIS(600),ELOAD
     +     (50,24),NOFIX(40),
     +     IFPRE(40,3),LNODS(50,8),MATNO
     +     (50)
      COMMON/WORK/ELCOD(2,8),SHAPE(8),
     +     DERIV(2,8),DMATX(5,5),CARTD
     +     (2,8),DBMAT(5,24),BMATX(5,
     +     24),SMATX(5,24,4),POSGP(2),
     +     WEIGP(2),GPCOD(2,4),NEROR(24)
c
      common / load  / unimoment(20,2),tarshape(200)
      common / HeatLine / itnl,coef(25,24,20,2)
      common / ga10  / differsum
c
cccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
c
c     This is a Heat Coditions Determination Program version 0.1
c     started written on August 22, 1996 by Hideki Shimizu
c
c     updated on April 24, 1997
c
c
cccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
c       PROGRAM BOK1
C     +(INPUT,OUTPUT,TAPE5=INPUT,
C     +TAPE6=OUTPUT,TAPE1,TAPE2,TAPE3,TAPE4)
C
c
      open(unit=5, file='data.dat', access='sequential',
     +status='old')
      open(unit=6, file='output.out', access='sequential',
```

```
     +status='unknow')
      open(unit=1, form='unformatted', access='sequential',
     +recl=10000, status='scratch')
      open(unit=2, form='unformatted', access='sequential',
     +recl=10000, status='scratch')
      open(unit=3, form='unformatted', access='sequential',
     +recl=10000, status='scratch')
      open(unit=4, form='unformatted', access='sequential',
     +recl=10000, status='scratch')
C
      READ(5,900) NPROB
 900  FORMAT(I5)
      WRITE(6,905) NPROB
 905  FORMAT(1H0,5X,
     +23HTOTAL NO. OF PROBLEMS =,I5)
c      DO 20 IPROB=1,NPROB
         rewind 6
C         REWIND 1
C         REWIND 2
C         REWIND 3
C         REWIND 4
         READ(5,910) TITLE
 910     FORMAT(18A4)
         WRITE(6,915) IPROB,TITLE
 915     FORMAT(/////,6X,12HPROBLEM NO. ,I3,10X,
     +18A4)
C
C*** CALL THE SUBROUTINE WHICH READS MOST OF
C    THE PROBLEM DATA
C
         CALL INPUT
C      write(6,1115) (nofix(i),i=1,nvfix)
C 1115 format(19hAfter Input nofix =,15i4)
C
C*** NEXT CREATE THE ELEMENT STIFFNESS FILE.
C
         CALL STIFPB
         icase=1
C
C*** COMPUTE LOADS, AFTER READING THE RELEVANT
C    EXTRA DATA
C
C      write(6,1116) (nofix(i),i=1,nvfix)
C 1116 format(21hBefore LOADPB nofix =,15i4)
c
c  Call the input subroutine.
c      TIME0=SECNDS(0.0)
      call GA_input
c
c  Perform necessary initialization and read the ga.restart file.
      call initial(istart,npossum,ig2sum)
c
c  $$$$$ Main generational processing loop. $$$$$
      kount=0
      do 20 i=istart,maxgen+istart-1
         write (6,1111) i
```

```
c         write (24,1111) i
         write(6,1050)
c
c  Evaluate the population, assign fitness, establish the best
c  individual, and write output information.
         call evalout(iskip,iend,ibest)
         if(npopsiz.eq.1 .or. iskip.ne.0) then
            close(24)
            stop
         endif
c
c  Implement "niching".
         if (iniche.ne.0) call niche
c
c  Enter selection, crossover and mutation loop.
         ncross=0
         ipick=npopsiz
         do 45 j=1,npopsiz,nchild
c
c  Perform selection.
            call selectn(ipick,j,mate1,mate2)
c
c  Now perform crossover between the randomly selected pair.
            call crosovr(ncross,j,mate1,mate2)
 45      continue
         write(6,1225) ncross
c         write(24,1225) ncross
c
c  Now perform random mutations.  If running micro-GA, skip mutation.
         if (microga.eq.0) call mutate
c
c  Write child array back into parent array for new generation.  Check
c  to see if the best parent was replicated.
         call newgen(ielite,npossum,ig2sum,ibest)
c
c  Implement micro-GA if enabled.
         if (microga.ne.0) call gamicro(i,npossum,ig2sum,ibest)
c
c  Write to restart file.
         call restart(i,istart,kount)
c         CPU = SECNDS(TIME0)
c         write(6,1400) cpu
c         write(24,1400) cpu
 20   continue
c  $$$$$ End of main generational processing loop. $$$$$
c 999  continue
      CLOSE (24)
c
 1050 format(1x,' #      Binary Code',8x,'Param1 Param2 Param3',
     +          ' Param4 Param5 Param6 Param7 Param8 Fitness')
 1111 format(//'#################  Generation',i5,'  #################')
 1225 format(/'  Number of Crossovers      =',i5)
c 1400 format(2x,'CPU time for generation=',e10.4)
c
         close(unit=5)
         close(unit=6)
```

```
         close(unit=1)
         close(unit=2)
         close(unit=3)
         close(unit=4)
C
         STOP
         END
c#######################################################################
      subroutine GA_input
c
c  This subroutine inputs information from the ga.inp file.
c
      implicit double precision (a-h,o-z)
      save
c
      include 'params.f'
      dimension nposibl(nparmax),nichflg(nparmax)
      dimension parmax(nparmax),parmin(nparmax),pardel(nparmax)
c
      common / ga1   / npopsiz,nowrite
      common / ga2   / nparam,nchrome
      common / ga6   / parmax,parmin,pardel,nposibl
      common / ga8   / nichflg
      common /inputga/ pcross,pmutate,pcreep,maxgen,idum,irestrt,
     +                 itourny,ielite,icreep,iunifrm,iniche,
     +                 iskip,iend,nchild,microga,kountmx
c
      namelist / ga   / irestrt,npopsiz,pmutate,maxgen,idum,pcross,
     +                  itourny,ielite,icreep,pcreep,iunifrm,iniche,
     +                  iskip,iend,nchild,nparam,parmin,parmax,nposibl,
     +                  nowrite,nichflg,microga,kountmx
c
      kountmx=5
      irestrt=0
      itourny=0
      ielite=0
      iunifrm=0
      iniche=0
      iskip=0
      iend=0
      nchild=1
      do 2 i=1,nparam
         nichflg(i)=1
 2    continue
      microga=0
c
      OPEN (UNIT=24, FILE='ga.out', STATUS='UNKNOWN')
      rewind 24
      OPEN (UNIT=23, FILE='ga.inp', STATUS='OLD')
      READ (23, NML = ga)
      CLOSE (23)
      itourny=1
c      if (itourny.eq.0) nchild=2
c
c  Check for array sizing errors.
      if (npopsiz.gt.indmax) then
```

```
         write(6,1600) npopsiz
c         write(24,1600) npopsiz
c         close(24)
         stop
      endif
      if (nparam.gt.nparmax) then
         write(6,1700) nparam
c         write(24,1700) nparam
c         close(24)
         stop
      endif
c
c  If using the microga option, reset some input variables
      if (microga.ne.0) then
         pmutate=0.0
         pcreep=0.0
         itourny=1
         ielite=1
         iniche=0
         nchild=1
         if (iunifrm.eq.0) then
            pcross=1.0
         else
            pcross=0.5
         endif
      endif
c
 1600 format(1x,'ERROR: npopsiz > indmax.  Set indmax = ',i6)
 1700 format(1x,'ERROR: nparam > nparmax.  Set nparmax = ',i6)
c
      return
      end
c
c#######################################################################
      subroutine initial(istart,npossum,ig2sum)
c
c  This subroutine sets up the program by generating the g0, g1 and
c  ig2 arrays, and counting the number of chromosomes required for the
c  specified input.  The subroutine also initializes the random number
c  generator, parent and iparent arrays (reads the ga.restart file).
      implicit double precision (a-h,o-z)
      save
c
      include 'params.f'
      dimension parent(indmax,nparmax),iparent(indmax,nchrmax)
      dimension nposibl(nparmax)
      dimension g0(nparmax),g1(nparmax),ig2(nparmax)
      dimension parmax(nparmax),parmin(nparmax),pardel(nparmax)
c
      common / ga1   / npopsiz,nowrite
      common / ga2   / nparam,nchrome
      common / ga3   / parent,iparent
      common / ga5   / g0,g1,ig2
      common / ga6   / parmax,parmin,pardel,nposibl
      common /inputga/ pcross,pmutate,pcreep,maxgen,idum,irestrt,
     +                 itourny,ielite,icreep,iunifrm,iniche,
```

```
     +                 iskip,iend,nchild,microga,kountmx
c
c
      do 3 i=1,nparam
         g0(i)=parmin(i)
         pardel(i)=parmax(i)-parmin(i)
         g1(i)=pardel(i)/dble(nposibl(i)-1)
 3    continue
      do 6 i=1,nparam
         do 7 j=1,40
            n2j=2**j
            if (n2j.ge.nposibl(i)) then
               ig2(i)=j
               goto 8
            endif
            if (j.ge.40) then
               write(6,2000)
c               write(24,2000)
c               close(24)
               stop
            endif
 7       continue
 8       continue
 6    continue
c
c  Count the total number of chromosomes (bits) required
      nchrome=0
      npossum=0
      ig2sum=0
      do 9 i=1,nparam
         nchrome=nchrome+ig2(i)
         npossum=npossum+nposibl(i)
         ig2sum=ig2sum+(2**ig2(i))
 9    continue
      if (nchrome.gt.nchrmax) then
         write(6,1800) nchrome
c         write(24,1800) nchrome
c         close(24)
         stop
      endif
c
      if (npossum.lt.ig2sum .and. microga.ne.0) then
         write(6,2100)
c         write(24,2100)
      endif
c
c  Initialize random number generator
      call ran3(idum,rand)
c
      if(irestrt.eq.0) then
c  Initialize the random distribution of parameters in the individual
c  parents when irestrt=0.
         istart=1
         do 10 i=1,npopsiz
            do 15 j=1,nchrome
               call ran3(1,rand)
```

```
                iparent(i,j)=1
                if(rand.lt.0.5) iparent(i,j)=0
 15          continue
 10       continue
          if (npossum.lt.ig2sum) call possibl(parent,iparent)
       else
c  If irestrt.ne.0, read from restart file.
          OPEN (UNIT=25, FILE='ga.restart', STATUS='OLD')
          rewind 25
          read(25,*) istart,npopsiz
          do 1 j=1,npopsiz
             read(25,*) k,(iparent(j,l),l=1,nchrome)
 1        continue
          CLOSE (25)
       endif
c
       if(irestrt.ne.0) call ran3(idum-istart,rand)
c
 1800 format(1x,'ERROR: nchrome > nchrmax.  Set nchrmax = ',i6)
 2000 format(1x,'ERROR: You have a parameter with a number of '/
     +       1x,'   possibilities > 2**40!  If you really desire this,'/
     +       1x,'   change the DO loop 7 statement and recompile.')
 2100 format(1x,'WARNING: for some cases, a considerable performance'/
     +       1x,'   reduction has been observed when running a non-'/
     +       1x,'   optimal number of bits with the micro-GA.'/
     +       1x,'   If possible, use values for nposibl of 2**n,'/
     +       1x,'   e.g. 2, 4, 8, 16, 32, 64, etc.  See ReadMe file.')
c
       return
       end
c
c#######################################################################
       subroutine evalout(iskip,iend,ibest)
c
c  This subroutine evaluates the population, assigns fitness,
c  establishes the best individual, and outputs information.
       implicit double precision (a-h,o-z)
       save
c
       include 'params.f'
       dimension parent(indmax,nparmax),iparent(indmax,nchrmax)
       dimension fitness(indmax)
       dimension paramsm(nparmax),paramav(nparmax),ibest(nchrmax)
c
       common / ga1   / npopsiz,nowrite
       common / ga2   / nparam,nchrome
       common / ga3   / parent,iparent
       common / ga4   / fitness
       common / load  / unimoment(20,2),tarshape(200)
       common / HeatLine / itnl,coef(25,24,20,2)
c
       fitsum=0.0
       best=0.0
       do 29 n=1,nparam
          paramsm(n)=0.0
 29    continue
```

```
      jstart=1
      jend=npopsiz
      if(iskip.ne.0) jstart=iskip
      if(iend.ne.0) jend=iend
      do 30 j=jstart,jend
         call decode(j,parent,iparent)
         if(iskip.ne.0 .and. iend.ne.0 .and. iskip.eq.iend)
     +   write(6,1075) j,(iparent(j,k),k=1,nchrome),
     +                  (parent(j,kk),kk=1,nparam),0.0
c
c  Call function evaluator, write out individual and fitness, and add
c  to the summation for later averaging.
c
         call func(j,funcval)
         fitness(j)=funcval
         write(6,1075) j,(iparent(j,k),k=1,nchrome),
     +                 (parent(j,kk),kk=1,nparam),fitness(j)
c         write(24,1075) j,(iparent(j,k),k=1,nchrome),
c     +                  (parent(j,kk),kk=1,nparam),fitness(j)
         fitsum=fitsum+fitness(j)
         do 22 n=1,nparam
            paramsm(n)=paramsm(n)+parent(j,n)
 22      continue
c
c  Check to see if fitness of individual j is the best fitness.
         if (fitness(j).gt.best) then
            best=fitness(j)
            jbest=j
            do 24 k=1,nchrome
               ibest(k)=iparent(j,k)
 24         continue
         endif
 30   continue
c
c  Compute parameter and fitness averages.
      fbar=fitsum/dble(npopsiz)
      do 23 n=1,nparam
         paramav(n)=paramsm(n)/dble(npopsiz)
 23   continue
c
c  Write output information
      if (npopsiz.eq.1) then
         write(6,1075) 1,(iparent(1,k),k=1,nchrome),
     +                 (parent(1,k),k=1,nparam),fitness(1)
         write(6,*) ' Average Values:'
         write(6,1275) (parent(1,k),k=1,nparam),fbar
c         write(24,1075) 1,(iparent(1,k),k=1,nchrome),
c     +                  (parent(1,k),k=1,nparam),fitness(1)
c         write(24,*) ' Average Values:'
c         write(24,1275) (parent(1,k),k=1,nparam),fbar
      else
         write(6,1275) (paramav(k),k=1,nparam),fbar
c         write(24,1275) (paramav(k),k=1,nparam),fbar
      endif
      write(6,1100) fbar
c      write(24,1100) fbar
```

```
      write(6,1200) best
c      write(24,1200) best
c
 1075 format(i3,1x,20i1,10(1x,f6.2),1x,f20.10)
 1100 format(1x,'Average Function Value of Generation=',f20.10)
 1200 format(1x,'Maximum Function Value              =',f20.10/)
 1275 format(/' Average Values:',10x,10(1x,f6.2),1x,f20.10/)
      return
      end
c
c#######################################################################
      subroutine niche
c
c  Implement "niching" through Goldberg's multidimensional phenotypic
c  sharing scheme with a triangular sharing function.  To find the
c  multidimensional distance from the best individual, normalize all
c  parameter differences.
c
      implicit double precision (a-h,o-z)
      save
c
      include 'params.f'
      dimension parent(indmax,nparmax),iparent(indmax,nchrmax)
      dimension fitness(indmax),nposibl(nparmax),nichflg(nparmax)
      dimension parmax(nparmax),parmin(nparmax),pardel(nparmax)
c
      common / ga1   / npopsiz,nowrite
      common / ga2   / nparam,nchrome
      common / ga3   / parent,iparent
      common / ga4   / fitness
      common / ga6   / parmax,parmin,pardel,nposibl
      common / ga8   / nichflg
c
      sigshar=0.0
      do 33 jj=1,nparam
         sigshar=sigshar+dble(nichflg(jj))
 33   continue
      if (sigshar.eq.0.0) then
         write(6,1900)
c         write(24,1900)
c         close(24)
         stop
      endif
      do 34 ii=1,npopsiz
         sumshar=0.0
         do 35 j=1,npopsiz
            del2=0.0
            do 36 k=1,nparam
               if (nichflg(k).ne.0) then
                  del2=del2+((parent(j,k)-parent(ii,k))/pardel(k))**2.0
               endif
 36         continue
            del=dsqrt(del2)
            delta=del/sigshar
            sumshar=sumshar+1-delta
 35      continue
```

```
         fitness(ii)=fitness(ii)/sumshar
 34   continue
c
 1900 format(1x,'ERROR: iniche=1 and all values in nichflg array = 0'/
     +       1x,'       Do you want to niche or not?')
c
      return
      end
c
c#######################################################################
      subroutine selectn(ipick,j,mate1,mate2)
c
c  Subroutine for selection operator.  Presently, tournament selection
c  is the only option available.
c
      implicit double precision (a-h,o-z)
      save
c
      include 'params.f'
      dimension parent(indmax,nparmax),child(indmax,nparmax)
      dimension fitness(indmax)
      dimension iparent(indmax,nchrmax),ichild(indmax,nchrmax)
c
      common / ga1   / npopsiz,nowrite
      common / ga2   / nparam,nchrome
      common / ga3   / parent,iparent
      common / ga4   / fitness
      common / ga7   / child,ichild
      common /inputga/ pcross,pmutate,pcreep,maxgen,idum,irestrt,
     +                 itourny,ielite,icreep,iunifrm,iniche,
     +                 iskip,iend,nchild,microga,kountmx
c
c  If tournament selection is chosen (i.e. itourny=1), then
c  implement "tournament" selection for selection of new population.
      if(itourny.eq.1) then
         call select(mate1,ipick)
         call select(mate2,ipick)
c        write(3,*) mate1,mate2,fitness(mate1),fitness(mate2)
         do 46 n=1,nchrome
            ichild(j,n)=iparent(mate1,n)
            if(nchild.eq.2) ichild(j+1,n)=iparent(mate2,n)
 46      continue
      endif
c
      return
      end
c
c#######################################################################
      subroutine crosovr(ncross,j,mate1,mate2)
c
c  Subroutine for crossover between the randomly selected pair.
      implicit double precision (a-h,o-z)
      save
c
      include 'params.f'
      dimension parent(indmax,nparmax),child(indmax,nparmax)
```

```
      dimension iparent(indmax,nchrmax),ichild(indmax,nchrmax)
c
      common / ga2   / nparam,nchrome
      common / ga3   / parent,iparent
      common / ga7   / child,ichild
      common /inputga/ pcross,pmutate,pcreep,maxgen,idum,irestrt,
     +                 itourny,ielite,icreep,iunifrm,iniche,
     +                 iskip,iend,nchild,microga,kountmx
c
      if (iunifrm.eq.0) then
c  Single-point crossover at a random chromosome point.
         call ran3(1,rand)
         if(rand.gt.pcross) goto 69
         ncross=ncross+1
         call ran3(1,rand)
         icross=2+dint(dble(nchrome-1)*rand)
         do 50 n=icross,nchrome
            ichild(j,n)=iparent(mate2,n)
            if(nchild.eq.2) ichild(j+1,n)=iparent(mate1,n)
 50      continue
      else
c  Perform uniform crossover between the randomly selected pair.
         do 60 n=1,nchrome
            call ran3(1,rand)
            if(rand.le.pcross) then
               ncross=ncross+1
               ichild(j,n)=iparent(mate2,n)
               if(nchild.eq.2) ichild(j+1,n)=iparent(mate1,n)
            endif
 60      continue
      endif
 69   continue
c
      return
      end
c
c#######################################################################
      subroutine mutate
c
      implicit double precision (a-h,o-z)
      save
c
      include 'params.f'
      dimension nposibl(nparmax)
      dimension child(indmax,nparmax),ichild(indmax,nchrmax)
      dimension g0(nparmax),g1(nparmax),ig2(nparmax)
      dimension parmax(nparmax),parmin(nparmax),pardel(nparmax)
c
      common / ga1   / npopsiz,nowrite
      common / ga2   / nparam,nchrome
      common / ga5   / g0,g1,ig2
      common / ga6   / parmax,parmin,pardel,nposibl
      common / ga7   / child,ichild
      common /inputga/ pcross,pmutate,pcreep,maxgen,idum,irestrt,
     +                 itourny,ielite,icreep,iunifrm,iniche,
     +                 iskip,iend,nchild,microga,kountmx
```

```
c
c  This subroutine performs mutations on the children generation.
c  Perform random jump mutation if a random number is less than pmutate.
c  Perform random creep mutation if a different random number is less
c  than pcreep.
      nmutate=0
      ncreep=0
      do 70 j=1,npopsiz
         do 75 k=1,nchrome
c  Jump mutation
            call ran3(1,rand)
            if (rand.le.pmutate) then
               nmutate=nmutate+1
               if(ichild(j,k).eq.0) then
                  ichild(j,k)=1
               else
                  ichild(j,k)=0
               endif
               if (nowrite.eq.0) write(6,1300) j,k
c               if (nowrite.eq.0) write(24,1300) j,k
            endif
 75      continue
c  Creep mutation (one discrete position away).
         if (icreep.ne.0) then
            do 76 k=1,nparam
               call ran3(1,rand)
               if(rand.le.pcreep) then
                  call decode(j,child,ichild)
                  ncreep=ncreep+1
                  creep=1.0
                  call ran3(1,rand)
                  if (rand.lt.0.5) creep=-1.0
                  child(j,k)=child(j,k)+g1(k)*creep
                  if (child(j,k).gt.parmax(k)) then
                     child(j,k)=parmax(k)-1.0*g1(k)
                  elseif (child(j,k).lt.parmin(k)) then
                     child(j,k)=parmin(k)+1.0*g1(k)
                  endif
                  call code(j,k,child,ichild)
                  if (nowrite.eq.0) write(6,1350) j,k
c                  if (nowrite.eq.0) write(24,1350) j,k
               endif
 76         continue
         endif
 70   continue
      write(6,1250) nmutate,ncreep
c      write(24,1250) nmutate,ncreep
c
 1250 format(/'  Number of Jump Mutations  =',i5/
     +        '  Number of Creep Mutations =',i5)
 1300 format('*** Jump mutation performed on individual  ',i4,
     +       ', chromosome ',i3,' ***')
 1350 format('*** Creep mutation performed on individual ',i4,
     +       ', parameter  ',i3,' ***')
c
      return
```

```
      end
c
c#######################################################################
      subroutine newgen(ielite,npossum,ig2sum,ibest)
c
c  Write child array back into parent array for new generation.  Check
c  to see if the best parent was replicated; if not, and if ielite=1,
c  then reproduce the best parent into a random slot.
c
      implicit double precision (a-h,o-z)
      save
c
      include 'params.f'
      dimension parent(indmax,nparmax),child(indmax,nparmax)
      dimension iparent(indmax,nchrmax),ichild(indmax,nchrmax)
      dimension ibest(nchrmax)
c
      common / ga1   / npopsiz,nowrite
      common / ga2   / nparam,nchrome
      common / ga3   / parent,iparent
      common / ga7   / child,ichild
c
      if (npossum.lt.ig2sum) call possibl(child,ichild)
      kelite=0
      do 94 j=1,npopsiz
         jelite=0
         do 95 n=1,nchrome
            iparent(j,n)=ichild(j,n)
            if (iparent(j,n).eq.ibest(n)) jelite=jelite+1
            if (jelite.eq.nchrome) kelite=1
 95      continue
 94   continue
      if (ielite.ne.0 .and. kelite.eq.0) then
         call ran3(1,rand)
         irand=1+dint(dble(npopsiz)*rand)
         do 96 n=1,nchrome
            iparent(irand,n)=ibest(n)
 96      continue
         write(6,1260) irand
c         write(24,1260) irand
      endif
c
 1260 format('  Elitist Reproduction on Individual ',i4)
c
      return
      end
c
c#######################################################################
      subroutine gamicro(i,npossum,ig2sum,ibest)
c
c  Micro-GA implementation subroutine
c
      implicit double precision (a-h,o-z)
      save
c
      include 'params.f'
```

```
      dimension parent(indmax,nparmax),iparent(indmax,nchrmax)
      dimension ibest(nchrmax)
c
      common / ga1   / npopsiz,nowrite
      common / ga2   / nparam,nchrome
      common / ga3   / parent,iparent
c
c  First, check for convergence of micro population.
c  If converged, start a new generation with best individual and fill
c  the remainder of the population with new randomly generated parents.
c
c  Count number of different bits from best member in micro-population
      icount=0
      do 81 j=1,npopsiz
         do 82 n=1,nchrome
            if(iparent(j,n).ne.ibest(n)) icount=icount+1
 82      continue
 81   continue
c
c  If icount less than 5% of number of bits, then consider population
c  to be converged.  Restart with best individual and random others.
      diffrac=dble(icount)/dble((npopsiz-1)*nchrome)
      if (diffrac.lt.0.05) then
      do 87 n=1,nchrome
         iparent(1,n)=ibest(n)
 87   continue
      do 88 j=2,npopsiz
         do 89 n=1,nchrome
            call ran3(1,rand)
            iparent(j,n)=1
            if(rand.lt.0.5) iparent(j,n)=0
 89      continue
 88   continue
      if (npossum.lt.ig2sum) call possibl(parent,iparent)
      write(6,1375) i
c      write(24,1375) i
      endif
c
 1375 format(//'%%%%%%%  Restart micro-population at generation',
     +       i5,'  %%%%%%%')
c
      return
      end
c
c#######################################################################
      subroutine select(mate,ipick)
c
c  This routine selects the better of two possible parents for mating.
c
      implicit double precision (a-h,o-z)
      save
c
      include 'params.f'
      common / ga1   / npopsiz,nowrite
      common / ga2   / nparam,nchrome
      common / ga3   / parent,iparent
```

```
      common / ga4   / fitness
      dimension parent(indmax,nparmax),iparent(indmax,nchrmax)
      dimension fitness(indmax)
c
      if(ipick+1.gt.npopsiz) call shuffle(ipick)
      ifirst=ipick
      isecond=ipick+1
      ipick=ipick+2
      if(fitness(ifirst).gt.fitness(isecond)) then
         mate=ifirst
      else
         mate=isecond
      endif
c     write(3,*)'select',ifirst,isecond,fitness(ifirst),fitness(isecond)
c
      return
      end
c
c#######################################################################
      subroutine shuffle(ipick)
c
c  This routine shuffles the parent array and its corresponding fitness
c
      implicit double precision (a-h,o-z)
      save
c
      include 'params.f'
      common / ga1   / npopsiz,nowrite
      common / ga2   / nparam,nchrome
      common / ga3   / parent,iparent
      common / ga4   / fitness
      dimension parent(indmax,nparmax),iparent(indmax,nchrmax)
      dimension fitness(indmax)
c
      ipick=1
      do 10 j=1,npopsiz-1
         call ran3(1,rand)
         iother=j+1+dint(dble(npopsiz-j)*rand)
         do 20 n=1,nchrome
            itemp=iparent(iother,n)
            iparent(iother,n)=iparent(j,n)
            iparent(j,n)=itemp
 20      continue
         temp=fitness(iother)
         fitness(iother)=fitness(j)
         fitness(j)=temp
 10   continue
c
      return
      end
c
c#######################################################################
      subroutine decode(i,array,iarray)
c
c  This routine decodes a binary string to a real number.
c
```

```
      implicit double precision (a-h,o-z)
      save
c
      include 'params.f'
      common / ga2   / nparam,nchrome
      common / ga5   / g0,g1,ig2
      dimension array(indmax,nparmax),iarray(indmax,nchrmax)
      dimension g0(nparmax),g1(nparmax),ig2(nparmax)
c
      l=1
      do 10 k=1,nparam
         iparam=0
         m=l
         do 20 j=m,m+ig2(k)-1
            l=l+1
            iparam=iparam+iarray(i,j)*(2**(m+ig2(k)-1-j))
 20      continue
         array(i,k)=g0(k)+g1(k)*dble(iparam)
 10   continue
c
      return
      end
c
c#######################################################################
      subroutine code(j,k,array,iarray)
c
c  This routine codes a parameter into a binary string.
c
      implicit double precision (a-h,o-z)
      save
c
      include 'params.f'
      common / ga2   / nparam,nchrome
      common / ga5   / g0,g1,ig2
      dimension array(indmax,nparmax),iarray(indmax,nchrmax)
      dimension g0(nparmax),g1(nparmax),ig2(nparmax)
c
c  First, establish the beginning location of the parameter string of
c  interest.
      istart=1
      do 10 i=1,k-1
         istart=istart+ig2(i)
 10   continue
c
c  Find the equivalent coded parameter value, and back out the binary
c  string by factors of two.
      m=ig2(k)-1
      if (g1(k).eq.0.0) return
      iparam=nint((array(j,k)-g0(k))/g1(k))
      do 20 i=istart,istart+ig2(k)-1
         iarray(j,i)=0
         if ((iparam+1).gt.(2**m)) then
            iarray(j,i)=1
            iparam=iparam-2**m
         endif
         m=m-1
```

```
 20   continue
c     write(3,*)array(j,k),iparam,(iarray(j,i),i=istart,istart+ig2(k)-1)
c
      return
      end
c
c#######################################################################
c
      subroutine possibl(array,iarray)
c
c  This subroutine determines whether or not all parameters are within
c  the specified range of possibility.  If not, the parameter is
c  randomly reassigned within the range.  This subroutine is only
c  necessary when the number of possibilities per parameter is not
c  optimized to be 2**n, i.e. if npossum < ig2sum.
c
      implicit double precision (a-h,o-z)
      save
c
      include 'params.f'
      common / ga1   / npopsiz,nowrite
      common / ga2   / nparam,nchrome
      common / ga5   / g0,g1,ig2
      common / ga6   / parmax,parmin,pardel,nposibl
      dimension array(indmax,nparmax),iarray(indmax,nchrmax)
      dimension g0(nparmax),g1(nparmax),ig2(nparmax),nposibl(nparmax)
      dimension parmax(nparmax),parmin(nparmax),pardel(nparmax)
c
      do 10 i=1,npopsiz
         call decode(i,array,iarray)
         do 20 j=1,nparam
            n2ig2j=2**ig2(j)
            if(nposibl(j).ne.n2ig2j .and. array(i,j).gt.parmax(j)) then
               call ran3(1,rand)
               irand=dint(dble(nposibl(j))*rand)
               array(i,j)=g0(j)+dble(irand)*g1(j)
               call code(i,j,array,iarray)
               if (nowrite.eq.0) write(6,1000) i,j
c               if (nowrite.eq.0) write(24,1000) i,j
            endif
 20      continue
 10   continue
c
 1000 format('*** Parameter adjustment to individual     ',i4,
     +       ', parameter  ',i3,' ***')
c
      return
      end
c
c#######################################################################
      subroutine restart(i,istart,kount)
c
c  This subroutine writes restart information to the ga.restart file.
c
      implicit double precision (a-h,o-z)
      save
```

```
c
      include 'params.f'
      common / ga1   / npopsiz,nowrite
      common / ga2   / nparam,nchrome
      common / ga3   / parent,iparent
      dimension parent(indmax,nparmax),iparent(indmax,nchrmax)
      common /inputga/ pcross,pmutate,pcreep,maxgen,idum,irestrt,
     +                 itourny,ielite,icreep,iunifrm,iniche,
     +                 iskip,iend,nchild,microga,kountmx
c
      kount=kount+1
      if(i.eq.maxgen+istart-1 .or. kount.eq.kountmx) then
         OPEN (UNIT=25, FILE='ga.restart', STATUS='OLD')
         rewind 25
         write(25,*) i+1,npopsiz
         do 80 j=1,npopsiz
            write(25,1500) j,(iparent(j,l),l=1,nchrome)
 80      continue
         CLOSE (25)
         kount=0
      endif
c
 1500 format(i5,3x,22i2)
c
      return
      end
c
c#######################################################################
      subroutine ran3(idum,rand)
c
c  Returns a uniform random deviate between 0.0 and 1.0.  Set idum to
c  any negative value to initialize or reinitialize the sequence.
c  This function is taken from W.H. Press', "Numerical Recipes" p. 199.
c
      implicit double precision (a-h,m,o-z)
      save
c      implicit real*4(m)
      parameter (mbig=4000000.,mseed=1618033.,mz=0.,fac=1./mbig)
c     parameter (mbig=1000000000,mseed=161803398,mz=0,fac=1./mbig)
c
c  According to Knuth, any large mbig, and any smaller (but still large)
c  mseed can be substituted for the above values.
      dimension ma(55)
      data iff /0/
      if (idum.lt.0 .or. iff.eq.0) then
         iff=1
         mj=mseed-dble(iabs(idum))
         mj=dmod(mj,mbig)
         ma(55)=mj
         mk=1
         do 11 i=1,54
            ii=mod(21*i,55)
            ma(ii)=mk
            mk=mj-mk
            if(mk.lt.mz) mk=mk+mbig
            mj=ma(ii)
```

```
 11       continue
          do 13 k=1,4
             do 12 i=1,55
                ma(i)=ma(i)-ma(1+mod(i+30,55))
                if(ma(i).lt.mz) ma(i)=ma(i)+mbig
 12          continue
 13       continue
          inext=0
          inextp=31
          idum=1
       endif
       inext=inext+1
       if(inext.eq.56) inext=1
       inextp=inextp+1
       if(inextp.eq.56) inextp=1
       mj=ma(inext)-ma(inextp)
       if(mj.lt.mz) mj=mj+mbig
       ma(inext)=mj
       rand=mj*fac
       return
       end
c
c#######################################################################
c
       subroutine func(j,funcval)
c
       implicit double precision (a-h,o-z)
       save
c
       include 'params.f'
       dimension parent(indmax,nparmax)
       dimension iparent(indmax,nchrmax)
c
       COMMON/CONTRO/NPOIN,NELEM,NNODE,NDOFN,NDIME,
     +     NSTRE,NTYPE,NGAUS,NPROP,NMATS,NVFIX,NEVAB,
     +     ICASE,NCASE,ITEMP,IPROB,NPROB
       COMMON/LGDATA/COORD(200,2),PROPS(10,
     +     4),PRESC(40,3),ASDIS(600),ELOAD
     +     (50,24),NOFIX(40),
     +     IFPRE(40,3),LNODS(50,8),MATNO
     +     (50)
       common / ga2   / nparam,nchrome
       common / ga3   / parent,iparent
       common / ga10  / differsum
       common / load  / unimoment(20,2),tarshape(200)
       common / HeatLine / itnl,coef(25,24,20,2)
c
c       write(6,10)
c 10    format(//,'++++++++++++These are parents+++++++++++++')
c       write(6,20) (parent(j,k),k=1,nparam)
c 20    format(/,'parents = ',15f10.5)
c
c       write(6,30) j
c 30    format(/,'j = ',i3)
c
       call LOADPB(j)
```

```
      call FRONT
      call compare
c     funcval=1/differsum
```

See Equation 5.1.

```
      funcval=1/(1+differsum/props(numat,3)/npoin)
c
cc      write(6,100)funcval
cc 100  format(/,'output from func  funcval= ',f15.5)
c
      return
      end
c#######################################################################
      subroutine compare
c
      implicit double precision (a-h,o-z)
      save
c
      COMMON/CONTRO/NPOIN,NELEM,NNODE,NDOFN,NDIME,
     +     NSTRE,NTYPE,NGAUS,NPROP,NMATS,NVFIX,NEVAB,
     +     ICASE,NCASE,ITEMP,IPROB,NPROB
      COMMON/LGDATA/COORD(200,2),PROPS(10,
     +     4),PRESC(40,3),ASDIS(600),ELOAD
     +     (50,24),NOFIX(40),
     +     IFPRE(40,3),LNODS(50,8),MATNO
     +     (50)
      common / ga10  / differsum
      common / load  / unimoment(20,2),tarshape(200)
c
      differ=0.0
      differsum=0.0
      do 10 ipoin=1,npoin
         ngish=ipoin*ndofn-ndofn+1
         differ=abs(tarshape(ipoin)-asdis(ngish))
         differsum=differsum+differ
 10   continue
c      write(6,100)differsum
c 100  format(/,'output from compare differsum= ',f15.5)
      return
      end
c#################################################################
      SUBROUTINE INPUT
C
      implicit double precision (a-h,o-z)
      save
C
      COMMON/CONTRO/NPOIN,NELEM,NNODE,NDOFN,NDIME,
     +     NSTRE,NTYPE,NGAUS,NPROP,NMATS,NVFIX,NEVAB,
     +     ICASE,NCASE,ITEMP,IPROB,NPROB
      COMMON/LGDATA/COORD(200,2),PROPS(10,
     +     4),PRESC(40,3),ASDIS(600),ELOAD
     +     (50,24),NOFIX(40),
     +     IFPRE(40,3),LNODS(50,8),MATNO
     +     (50)
      COMMON/WORK/ELCOD(2,8),SHAPE(8),
```

```
     +     DERIV(2,8),DMATX(5,5),CARTD
     +     (2,8),DBMAT(5,24),BMATX(5,
     +     24),SMATX(5,24,4),POSGP(2),
     +     WEIGP(2),GPCOD(2,4),NEROR(24)
c
      common / load / unimoment(20,2),tarshape(200)
      common / HeatLine / itnl,coef(25,24,20,2)
      dimension pointcoef(3)
c
      data pi /3.14159265359/
C
C*** READ THE FIRST DATA CARD, AND ECHO IT
C    IMMEDIATELY.
      READ(5,900) NPOIN,NELEM,NVFIX,NCASE,NTYPE,
     . NNODE,NDOFN,NMATS,NPROP,NGAUS,NDIME,NSTRE
  900 FORMAT(12I5)
      NEVAB=NDOFN*NNODE
      WRITE(6,905) NPOIN,NELEM,NVFIX,NCASE,NTYPE,
     . NNODE,NDOFN,NMATS,NPROP,NGAUS,NDIME,
     . NSTRE,NEVAB
  905 FORMAT(//8H NPOIN =,I4,4X,8H NELEM =,I4,
     . 4X,8H NVFIX =,I4,4X,8H NCASE =,I4,4X,
     . 8H NTYPE =,I4,4X,8H NNODE =,I4,4X,
     . 8H NDOFN =,I4// 8H NMATS =,I4,4X,
     . 8H NPROP =,I4,4X,8H NGAUS =,I4,4X,
     . 8H NDIME =,I4,4X,8H NSTRE =,I4,4X,
     . 8H NEVAB =,I4)
      CALL CHECK1
C
C*** READ THE ELEMENT NODAL CONNECTIONS, AND
C    THE PROPERTY NUMBERS.
C
      WRITE(6,910)
  910 FORMAT(//8H ELEMENT,3X,8HPROPERTY,6X,
     . 12HNODE NUMBERS)
      DO 10 IELEM=1,NELEM
      READ(5,900) NUMEL,MATNO(NUMEL),
     . (LNODS(NUMEL,INODE),INODE=1,NNODE)
   10 WRITE(6,915) NUMEL,MATNO(NUMEL),
     . (LNODS(NUMEL,INODE),INODE=1,NNODE)
  915 FORMAT(1X,I5,I9,6X,8I5)
C
C*** ZERO ALL THE NODAL COORDINATES, PRIOR
C    TO READING SOME OF THEM.
C
      DO 20 IPOIN=1,NPOIN
      DO 20 IDIME=1,NDIME
   20 COORD(IPOIN,IDIME)=0.0
C
C*** READ SOME NODAL COORDINATES, FINISHING
C    WITH THE LAST NODE OF ALL.
C
      WRITE(6,920)
  920 FORMAT(//25H  NODAL POINT COORDINATES)
      WRITE(6,925)
  925 FORMAT(6H  NODE,7X,1HX,9X,1HY)
```

```
   30 READ(5,930) IPOIN,(COORD(IPOIN,IDIME),
     . IDIME=1,NDIME)
  930 FORMAT(I5,5F10.5)
      IF(IPOIN.NE.NPOIN) GO TO 30
C
C*** INTERPOLATE COORDINATES OF MID-SIDE NODES
C
      IF(NDIME.EQ.1) GO TO 40
      CALL NODEXY
   40 CONTINUE
      DO 50 IPOIN=1,NPOIN
   50 WRITE(6,935) IPOIN,(COORD(IPOIN,IDIME),
     . IDIME=1,NDIME)
  935 FORMAT(1X,I5,3F10.3)
C
C*** READ THE FIXED VALUES.
C
      WRITE(6,940)
  940 FORMAT(//17H RESTRAINED NODES)
      WRITE(6,945)
  945 FORMAT(5H NODE,1X,4HCODE,6X,
     . 12HFIXED VALUES)
      IF(NDOFN.NE.2) GO TO 70
      DO 60 IVFIX=1,NVFIX
      READ(5,950) NOFIX(IVFIX),(IFPRE(IVFIX,
     . IDOFN),IDOFN=1,NDOFN),(PRESC(IVFIX,IDOFN),
     . IDOFN=1,NDOFN)
   60 WRITE(6,950) NOFIX(IVFIX),(IFPRE(IVFIX,
     . IDOFN),IDOFN=1,NDOFN),(PRESC(IVFIX,IDOFN),
     . IDOFN=1,NDOFN)
  950 FORMAT(1X,I4,3X,2I1,2F10.6)
      GO TO 90
   70 DO 80 IVFIX=1,NVFIX
      READ(5,955) NOFIX(IVFIX),(IFPRE(IVFIX,
     . IDOFN),IDOFN=1,NDOFN),(PRESC(IVFIX,IDOFN),
     . IDOFN=1,NDOFN)
   80 WRITE(6,955) NOFIX(IVFIX),(IFPRE(IVFIX,
     . IDOFN),IDOFN=1,NDOFN),(PRESC(IVFIX,IDOFN),
     . IDOFN=1,NDOFN)
  955 FORMAT(1X,I4,2X,3I1,3F10.6)
   90 CONTINUE
C
C*** READ THE AVAILABLE SELECTION OF ELEMENT
C    PROPERTIES.
C
      WRITE(6,960)
  960 FORMAT(//21H  MATERIAL PROPERTIES)
      WRITE(6,965)
  965 FORMAT(8H  NUMBER,7X,10HPROPERTIES)
      DO 100 IMATS=1,NMATS
      READ(5,930) NUMAT,(PROPS(NUMAT,IPROP),
     . IPROP=1,NPROP)
  100 WRITE(6,970) NUMAT,(PROPS(NUMAT,IPROP),
     . IPROP=1,NPROP)
  970 FORMAT(1X,I5,7X,5E14.6)
c
```

```
c*** read the magnitude for the moment
c
c       read(5,*)unimoment
c       write(6,*)unimoment
c
c start of resivion on 10/08/96
c*** read the distance from a heating line and selection width
       read(5,*) dista, sw
       write(6,*) dista,sw
c start of resivion on 10/12/96
c*** read the angular distortion for a standard heat process
c     thetadeg is in degree; standardwidth is in mm
       read(5,*)thetadeg,standwidth
       write(6,*)thetadeg,standwidth
c     tranform theta into radian
       theta=thetadeg*pi/180.0
```

See Equation 5.5.

```
c standard total moment per unit length: stmpu
       stmpu=(theta*props(1,1)*props(1,3)**3)/(24.0*dista)
c*** read the number of heating lines:itnl
       read(5,*) itnl
       write(6,*) itnl
c*** read sets of points on heating lines
       do 1 inl=1,itnl
       read(5,*) ln,x_1,y_1,x_2,y_2
       write(6,*) ln,x_1,y_1,x_2,y_2
c*** calculate Heat Line Length:hll
       hll=sqrt((x_2-x_1)**2+(y_2-y_1)**2)
```

See Equation 5.6.

```
c*** total moment for the heating line:tm
       tm=hll*stmpu
       write(6,5)tm
 5     format(/,'tm = ',f20.5)
c*** set counters zero to count the number of selected noes
       icountposi=0
       icountnega=0
c end of revision on 10/12/96
c*** input the range of applying moments
          do 2 ipoin=1,npoin
             xn=coord(ipoin,1)
             yn=coord(ipoin,2)
c             write(6,1001)x_1,y_1,x_2,y_2,xn,yn
c 1001        format('x_1 =',f8.4,'y_1 =',f8.4,'x_2 =',f8.4,'y_2 =',
c     +              f8.4,'xn = ',f8.4,'yn = ',f8.4)
```

See Equations 5.8 and 5.9.

```
             numofd=-(y_2-y_1)*xn+(x_2-x_1)*yn+(y_2-y_1)*x_1-(x_2-
     +               x_1)*y_1
             dist12=sqrt((y_2-y_1)**2+(x_2-x_1)**2)
             distance=abs(numofd/dist12)
```

```
            halfsw=sw/2.0
            gthresh=dista+halfsw
            sthresh=dista-halfsw
c
c            write(6,1000)distance,gthresh,sthresh
c 1000       format('distance = ',f8.4,'gthresh = ',f8.4
c     +             'sthresh = ',f8.4)
c
      if ((distance.le.gthresh).and.(distance.ge.sthresh)) then
         lodpt=ipoin
```

See Equation 5.12.

```
         if((x_2-x_1).eq.0.0)then
            alfa=pi/2.0
         else
            alfa=atan((y_2-y_1)/(x_2-x_1))
         endif
         write(6,1002)alfa
 1002    format('alfa = ',f10.5)
```

See Equations 5.10 and 5.11.

```
         pointcoef(1)=0.0
         pointcoef(2)=sin(alfa)
         pointcoef(3)=-cos(alfa)
         do 1100 ielem=1,nelem
            do 1100 inode=1,nnode
               nloca=lnods(ielem,inode)
c               write(6,1003)lodpt,ielem,inode,nloca
c 1003          format('lodpt =',i5,'  ielem =',i5,'  inode =',i5,
c     +                '  nloca =',i5)
               if (lodpt.eq.nloca) go to 1101
 1100       continue
 1101       do 1102 idofn=1,ndofn
                  ngash=(inode-1)*ndofn+idofn
                  if(numofd.gt.0.0) then
                     coef(ielem,ngash,inl,1)=pointcoef(idofn)
                     icountposi=icountposi+1
c                  write(6,11)ielem,inode,nloca,
c     +            coef(ielem,ngash,inl,1)
c 11               format('ielem=',i4,' inode=',i3,' nloca=',i4,
c     +            ' coef(ielem,ngash,inl,1)=',f15.5)
                  else
                     coef(ielem,ngash,inl,2)=-pointcoef(idofn)
                     icountnega=icountnega+1
c                  write(6,12)ielem,inode,nloca,
c     +            coef(ielem,ngash,inl,2)
c 12               format('ielem=',i4,' inode=',i3,' nloca=',i4,
c     +            ' coef(ielem,ngash,inl,2)=',f15.5)
                  endif
 1102        continue
      endif
c         write(6,3)inl,ielem,ngash,coef(ielem,ngash,inl,1),
c     +             coef(ielem,ngash,inl,2)
c 3       format(' inl=',i3,' ielem=',i5,' ngash=',i5,' coef=',f10.5,
```

```
c     +          'for posi ',f10.5,'for nega')
 2    continue
c start of revision on 10/12/96
c*** eliminate double count
      icount_p=icountposi/ndofn
      icount_n=icountnega/ndofn
```

See Equation 5.7.

```
c*** calculate unitmoment for one DOF of a node
c    for positive moment
      unimoment(inl,1)=tm/icount_p
c    for negative moment
      unimoment(inl,2)=tm/icount_n
      write(6,4) inl,icount_p,icount_n,
     +           (unimoment(inl,isign),isign=1,2)
 4    format('inl=',i3,' icount_p=',i3,' icount_n=',i3,
     +             ' unimoment=',f15.5,3x,f15.5)
c end of revision on 10/12/96
 1    continue
c end of revision on 10/08/96
C
C*** SET UP GAUSSIAN INTEGRATION CONSTANTS
C
      CALL GAUSSQ
      CALL CHECK2
c
c*** read the target shape of the plate
c
      do 200 ipoin=1,npoin
         read(5,*) ii,tarshape(ii)
         write(6,*) ii,tarshape(ii)
 200  continue
c
      RETURN
      END
C*******************************************
      SUBROUTINE NODEXY
C
      implicit double precision (a-h,o-z)
      save
c
      COMMON/CONTRO/NPOIN,NELEM,NNODE,NDOFN,NDIME,
     +     NSTRE,NTYPE,NGAUS,NPROP,NMATS,NVFIX,NEVAB,
     +     ICASE,NCASE,ITEMP,IPROB,NPROB
      COMMON/LGDATA/COORD(200,2),PROPS(10,
     +     4),PRESC(40,3),ASDIS(600),ELOAD
     +     (50,24),NOFIX(40),
     +     IFPRE(40,3),LNODS(50,8),MATNO
     +     (50)
      COMMON/WORK/ELCOD(2,8),SHAPE(8),
     +     DERIV(2,8),DMATX(5,5),CARTD
     +     (2,8),DBMAT(5,24),BMATX(5,
     +     24),SMATX(5,24,4),POSGP(2),
     +     WEIGP(2),GPCOD(2,4),NEROR(24)
C
```

```
C*** LOOP OVER EACH EMELENT
C
      DO 30 IELEM=1,NELEM
C
C*** LOOP OVER EACH ELEMENT EDGE
C
      DO 20 INODE=1,NNODE,2
C
C*** COMPUTE THE NODE NUMBER OF THE FIRST NODE
C
      NODST=LNODS(IELEM,INODE)
      IGASH=INODE+2
      IF(IGASH.GT.NNODE) IGASH=1
C
C*** COMPUTE THE NODE NUMBER OF THE LAST NODE
C
      NODFN=LNODS(IELEM,IGASH)
      MIDPT=INODE+1
C
C*** COMPUTE THE NODE NUMBER OF THE
C    INTERMEDIATE NODE
C
      NODMD=LNODS(IELEM,MIDPT)
      TOTAL=ABS(COORD(NODMD,1))+
     . ABS(COORD(NODMD,2))
C
C*** IF THE COORDINATES OF THE INTERMEDIATE
C    NODE ARE BOTH ZERO INTERPOLATE BY A
C    STRAIGHT LINE
C
      IF(TOTAL.GT.0.0) GO TO 20
      KOUNT=1
   10 COORD(NODMD,KOUNT)=(COORD(NODST,KOUNT)+
     . COORD(NODFN,KOUNT))/2.0
      KOUNT=KOUNT+1
      IF(KOUNT.EQ.2) GO TO 10
   20 CONTINUE
   30 CONTINUE
      RETURN
      END
C*****************************************
      SUBROUTINE GAUSSQ
C
      implicit double precision (a-h,o-z)
      save
c
      COMMON/CONTRO/NPOIN,NELEM,NNODE,NDOFN,NDIME,
     +     NSTRE,NTYPE,NGAUS,NPROP,NMATS,NVFIX,NEVAB,
     +     ICASE,NCASE,ITEMP,IPROB,NPROB
      COMMON/LGDATA/COORD(200,2),PROPS(10,
     +     4),PRESC(40,3),ASDIS(600),ELOAD
     +     (50,24),NOFIX(40),
     +     IFPRE(40,3),LNODS(50,8),MATNO
     +     (50)
      COMMON/WORK/ELCOD(2,8),SHAPE(8),
     +     DERIV(2,8),DMATX(5,5),CARTD
```

```
     +      (2,8),DBMAT(5,24),BMATX(5,
     +      24),SMATX(5,24,4),POSGP(2),
     +      WEIGP(2),GPCOD(2,4),NEROR(24)
C
      IF(NGAUS.GT.2) GO TO 10
      POSGP(1)=-0.577350269189626
      WEIGP(1)=1.0
      GO TO 20
   10 POSGP(1)=-0.774596669241483
      POSGP(2)=0.0
      WEIGP(1)=0.555555555555556
      WEIGP(2)=0.888888888888889
   20 KGAUS=NGAUS/2
      DO 30 IGASH=1,KGAUS
      JGASH=NGAUS+1-IGASH
      POSGP(JGASH)=-POSGP(IGASH)
      WEIGP(JGASH)=WEIGP(IGASH)
   30 CONTINUE
      RETURN
      END
C*****************************************
      SUBROUTINE BMATPB
C
C***  CALCULATES STRAIN MATRIX B
C     FOR PLATE BENDING ELEMENT
c
      implicit double precision (a-h,o-z)
      save
c
      COMMON/CONTRO/NPOIN,NELEM,NNODE,NDOFN,NDIME,
     +      NSTRE,NTYPE,NGAUS,NPROP,NMATS,NVFIX,NEVAB,
     +      ICASE,NCASE,ITEMP,IPROB,NPROB
      COMMON/LGDATA/COORD(200,2),PROPS(10,
     +      4),PRESC(40,3),ASDIS(600),ELOAD
     +      (50,24),NOFIX(40),
     +      IFPRE(40,3),LNODS(50,8),MATNO
     +      (50)
      COMMON/WORK/ELCOD(2,8),SHAPE(8),
     +      DERIV(2,8),DMATX(5,5),CARTD
     +      (2,8),DBMAT(5,24),BMATX(5,
     +      24),SMATX(5,24,4),POSGP(2),
     +      WEIGP(2),GPCOD(2,4),NEROR(24)
C
      DO 10 ISTRE=1,NSTRE
      DO 10 IEVAB=1,NEVAB
      BMATX(ISTRE,IEVAB)=0.0
   10 CONTINUE
      JGASH=0
      DO 20 INODE=1,NNODE
      IGASH=JGASH+1
      BMATX(4,IGASH)=CARTD(1,INODE)
      BMATX(5,IGASH)=CARTD(2,INODE)
      IGASH=IGASH+1
      JGASH=IGASH+1
      BMATX(1,IGASH)=-CARTD(1,INODE)
      BMATX(3,IGASH)=-CARTD(2,INODE)
```

```
      BMATX(4,IGASH)=-SHAPE(INODE)
      BMATX(2,JGASH)=-CARTD(2,INODE)
      BMATX(3,JGASH)=-CARTD(1,INODE)
      BMATX(5,JGASH)=-SHAPE(INODE)
   20 CONTINUE
      RETURN
      END
C*********************************************
      SUBROUTINE MODPB(LPROP)
C
C*** CALCULATES MATRIX OF ELASTIC RIGIDITIES
C    FOR PLATE BENDING ELEMENT
C
      implicit double precision (a-h,o-z)
      save
c
      COMMON/CONTRO/NPOIN,NELEM,NNODE,NDOFN,NDIME,
     +     NSTRE,NTYPE,NGAUS,NPROP,NMATS,NVFIX,NEVAB,
     +     ICASE,NCASE,ITEMP,IPROB,NPROB
      COMMON/LGDATA/COORD(200,2),PROPS(10,
     +     4),PRESC(40,3),ASDIS(600),ELOAD
     +     (50,24),NOFIX(40),
     +     IFPRE(40,3),LNODS(50,8),MATNO
     +     (50)
      COMMON/WORK/ELCOD(2,8),SHAPE(8),
     +     DERIV(2,8),DMATX(5,5),CARTD
     +     (2,8),DBMAT(5,24),BMATX(5,
     +     24),SMATX(5,24,4),POSGP(2),
     +     WEIGP(2),GPCOD(2,4),NEROR(24)
C
      DO 10 ISTRE=1,NSTRE
      DO 10 JSTRE=1,NSTRE
      DMATX(ISTRE,JSTRE)=0.0
   10 CONTINUE
      YOUNG=PROPS(LPROP,1)
      POISS=PROPS(LPROP,2)
      THICK=PROPS(LPROP,3)
      DMATX(1,1)=YOUNG*THICK*THICK*THICK
     ./(12.0*(1.0-POISS*POISS))
      DMATX(1,2)=POISS*DMATX(1,1)
      DMATX(2,2)=DMATX(1,1)
      DMATX(2,1)=DMATX(1,2)
      DMATX(3,3)=(1.0-POISS)*DMATX(1,1)/2.0
      DMATX(4,4)=YOUNG*THICK/(2.4*(1.0+POISS))
      DMATX(5,5)=DMATX(4,4)
      RETURN
      END
C*********************************************
      SUBROUTINE STIFPB
C
C***  CALCULATES ELEMENT STIFFNESS MATRIX
C     FOR PLATE BENDING ELEMENT
c
      implicit double precision (a-h,o-z)
      save
C
```

```
      DIMENSION ESTIF(24,24)
C
      COMMON/CONTRO/NPOIN,NELEM,NNODE,NDOFN,NDIME,
     +     NSTRE,NTYPE,NGAUS,NPROP,NMATS,NVFIX,NEVAB,
     +     ICASE,NCASE,ITEMP,IPROB,NPROB
      COMMON/LGDATA/COORD(200,2),PROPS(10,
     +     4),PRESC(40,3),ASDIS(600),ELOAD
     +     (50,24),NOFIX(40),
     +     IFPRE(40,3),LNODS(50,8),MATNO
     +     (50)
      COMMON/WORK/ELCOD(2,8),SHAPE(8),
     +     DERIV(2,8),DMATX(5,5),CARTD
     +     (2,8),DBMAT(5,24),BMATX(5,
     +     24),SMATX(5,24,4),POSGP(2),
     +     WEIGP(2),GPCOD(2,4),NEROR(24)
C
C*** LOOP OVER EACH ELEMENT
C
      DO 70 IELEM=1,NELEM
      LPROP=MATNO(IELEM)
C
C*** EVALUATE THE COORDINATES OF THE
C    ELEMENT NODAL POINTS
C
      DO 10 INODE=1,NNODE
      LNODE=LNODS(IELEM,INODE)
      DO 10 IDIME=1,NDIME
      ELCOD(IDIME,INODE)=COORD(LNODE,IDIME)
   10 CONTINUE
C
C*** INITIALIZE THE ELEMENT STIFFNESS MATRIX
C
      DO 20 IEVAB=1,NEVAB
      DO 20 JEVAB=1,NEVAB
      ESTIF(IEVAB,JEVAB)=0.0
   20 CONTINUE
C
C*** CALCULATE MATRIX OF ELASTIC RIGIDITIES
C
      CALL MODPB(LPROP)
      KGASP=0
C
C*** ENTER LOOPS FOR NUMERICAL INTEGRATION
C
      DO 50 IGAUS=1,NGAUS
      EXISP=POSGP(IGAUS)
      DO 50 JGAUS=1,NGAUS
      ETASP=POSGP(JGAUS)
      KGASP=KGASP+1
C
C*** EVALUATE THE SHAPE FUNCTIONS,
C    ELEMENT ARE, ETC.
C
      CALL SFR2(EXISP,ETASP)
      CALL JACOB2(IELEM,DJACB,KGASP)
      DAREA=DJACB*WEIGP(IGAUS)*WEIGP(JGAUS)
```

```
C
C*** EVAULATE THE B AND DB MATRICES
C
      CALL BMATPB
      CALL DBE
C
C*** CALCULATE THE ELEMENT STIFFNESS
C
      DO 30 IEVAB=1,NEVAB
      DO 30 JEVAB=IEVAB,NEVAB
      DO 30 ISTRE=1,NSTRE
      ESTIF(IEVAB,JEVAB)=ESTIF(IEVAB,JEVAB)+
     .BMATX(ISTRE,IEVAB)*DBMAT(ISTRE,JEVAB)
     .*DAREA
   30 CONTINUE
C
C*** STORE THE COMPONENTS OF THE DB MATRIX
C    FOR THE ELEMENT
C
      DO 40 ISTRE=1,NSTRE
      DO 40 IEVAB=1,NEVAB
      SMATX(ISTRE,IEVAB,KGASP)=
     .DBMAT(ISTRE,IEVAB)
   40 CONTINUE
   50 CONTINUE
C
C*** COSTRUCT THE LOWER TRIANGLE
C    OF THE STIFFNESS MATRIX
C
      DO 60 IEVAB=1,NEVAB
      DO 60 JEVAB=1,NEVAB
      ESTIF(JEVAB,IEVAB)=ESTIF(IEVAB,JEVAB)
   60 CONTINUE
C
C*** STORE THE STIFFNESS MATRIX,STRESS MATRIX,
C    AND SAMPLING POING COORDINATES FOR
C    EACH ELEMENT ON DISC FILE
C
      WRITE(1) ESTIF
      WRITE(3) SMATX,GPCOD
   70 CONTINUE
      RETURN
      END
C*********************************************
      SUBROUTINE SFR2(S,T)
C
C*** CALCULATES SHAPE FUNCTIONS AND THEIR
C    DERIVATIVES FOR 2D ELEMENTS
C
      implicit double precision (a-h,o-z)
      save
c
      COMMON/CONTRO/NPOIN,NELEM,NNODE,NDOFN,NDIME,
     +     NSTRE,NTYPE,NGAUS,NPROP,NMATS,NVFIX,NEVAB,
     +     ICASE,NCASE,ITEMP,IPROB,NPROB
      COMMON/LGDATA/COORD(200,2),PROPS(10,
```

```
     +     4),PRESC(40,3),ASDIS(600),ELOAD
     +     (50,24),NOFIX(40),
     +     IFPRE(40,3),LNODS(50,8),MATNO
     +     (50)
      COMMON/WORK/ELCOD(2,8),SHAPE(8),
     +     DERIV(2,8),DMATX(5,5),CARTD
     +     (2,8),DBMAT(5,24),BMATX(5,
     +     24),SMATX(5,24,4),POSGP(2),
     +     WEIGP(2),GPCOD(2,4),NEROR(24)
C
      S2=S*2.0
      T2=T*2.0
      SS=S*S
      TT=T*T
      ST=S*T
      SST=S*S*T
      STT=S*T*T
      ST2=S*T*2.0
C
C*** SHAPE FUCTIONS
C
      SHAPE(1)=(-1.0+ST+SS+TT-SST-STT)/4.0
      SHAPE(2)=(1.0-T-SS+SST)/2.0
      SHAPE(3)=(-1.0-ST+SS+TT-SST+STT)/4.0
      SHAPE(4)=(1.0+S-TT-STT)/2.0
      SHAPE(5)=(-1.0+ST+SS+TT+SST+STT)/4.0
      SHAPE(6)=(1.0+T-SS-SST)/2.0
      SHAPE(7)=(-1.0-ST+SS+TT+SST-STT)/4.0
      SHAPE(8)=(1.0-S-TT+STT)/2.0
C
C*** SHAPE FUNCTION DERIVATIVES
C
      DERIV(1,1)=(T+S2-ST2-TT)/4.0
      DERIV(1,2)=-S+ST
      DERIV(1,3)=(-T+S2-ST2+TT)/4.0
      DERIV(1,4)=(1.0-TT)/2.0
      DERIV(1,5)=(T+S2+ST2+TT)/4.0
      DERIV(1,6)=-S-ST
      DERIV(1,7)=(-T+S2+ST2-TT)/4.0
      DERIV(1,8)=(-1.0+TT)/2.0
      DERIV(2,1)=(S+T2-SS-ST2)/4.0
      DERIV(2,2)=(-1.0+SS)/2.0
      DERIV(2,3)=(-S+T2-SS+ST2)/4.0
      DERIV(2,4)=-T-ST
      DERIV(2,5)=(S+T2+SS+ST2)/4.0
      DERIV(2,6)=(1.0-SS)/2.0
      DERIV(2,7)=(-S+T2+SS-ST2)/4.0
      DERIV(2,8)=-T+ST
      RETURN
      END
C*************************************************
      SUBROUTINE JACOB2(IELEM,DJACB,KGASP)
C
C*** CALCULATE COORDINATES OF GAUSS POINTS
C    AND THE JACOBIAN MATRIX AND ITS DETERMINANT
C    AND THE INVERSE FOR 2D ELEMENTS
```

```
C
      implicit double precision (a-h,o-z)
      save
c
      DIMENSION XJACM(2,2),XJACI(2,2)
C
      COMMON/CONTRO/NPOIN,NELEM,NNODE,NDOFN,NDIME,
     +     NSTRE,NTYPE,NGAUS,NPROP,NMATS,NVFIX,NEVAB,
     +     ICASE,NCASE,ITEMP,IPROB,NPROB
      COMMON/LGDATA/COORD(200,2),PROPS(10,
     +     4),PRESC(40,3),ASDIS(600),ELOAD
     +     (50,24),NOFIX(40),
     +     IFPRE(40,3),LNODS(50,8),MATNO
     +     (50)
      COMMON/WORK/ELCOD(2,8),SHAPE(8),
     +     DERIV(2,8),DMATX(5,5),CARTD
     +     (2,8),DBMAT(5,24),BMATX(5,
     +     24),SMATX(5,24,4),POSGP(2),
     +     WEIGP(2),GPCOD(2,4),NEROR(24)
C
C
C*** CALCULATE COORDINATES OF SAMPLING POINT
C
      DO 10 IDIME=1,NDIME
         GPCOD(IDIME,KGASP)=0.0
      DO 10 INODE=1,NNODE
         GPCOD(IDIME,KGASP)=GPCOD(IDIME,KGASP)+
     +   ELCOD(IDIME,INODE)*SHAPE(INODE)
C         write(6,1000)idime,inode,elcod(idime,inode),
C     +    shape(inode),gpcod(idime,kgasp)
C 1000    format(8h idime =,i3,3x,8h inode =,i3,3x,
C     +   8h elcod =,f10.5,3x,8h shape =,f10.5,3x,
C     +   8h gpcod =,f10.5)
 10   CONTINUE
C
C*** CREATE JACOBIAN MATRIX XJACM
C
      DO 20 IDIME=1,NDIME
      DO 20 JDIME=1,NDIME
         XJACM(IDIME,JDIME)=0.0
         DO 20 INODE=1,NNODE
            XJACM(IDIME,JDIME)=XJACM(IDIME,JDIME)+
     +      DERIV(IDIME,INODE)*ELCOD(JDIME,INODE)
 20   CONTINUE
C
C*** CALCULATE DETERMINANT AND INVERSE OF
C    JACOBIAN MATRIX
C
      DJACB=XJACM(1,1)*XJACM(2,2)-XJACM(1,2)*
     +XJACM(2,1)
      IF(DJACB.GT.0.0) GO TO 30
      WRITE(6,900) IELEM
      STOP
 30   XJACI(1,1)=XJACM(2,2)/DJACB
      XJACI(2,2)=XJACM(1,1)/DJACB
      XJACI(1,2)=-XJACM(1,2)/DJACB
```

```
      XJACI(2,1)=-XJACM(2,1)/DJACB
C
C*** CALCULATE CARTESIAN DERIVATIVES
C
      DO 40 IDIME=1,NDIME
      DO 40 INODE=1,NNODE
         CARTD(IDIME,INODE)=0.0
      DO 40 JDIME=1,NDIME
         CARTD(IDIME,INODE)=CARTD(IDIME,INODE)+
     +   XJACI(IDIME,JDIME)*DERIV(JDIME,INODE)
 40   CONTINUE
 900  FORMAT(//,24HPROGRAM HALTED IN JACOB2,
     + /,11X,22H ZERO OR NEGATIVE AREA,/,
     + 10X,16H ELEMENT NUMBER ,I5)
      RETURN
      END
C*************************************************
      SUBROUTINE DBE
C
C*** CALCULATES D X B
C
      implicit double precision (a-h,o-z)
      save
c
      COMMON/CONTRO/NPOIN,NELEM,NNODE,NDOFN,NDIME,
     +     NSTRE,NTYPE,NGAUS,NPROP,NMATS,NVFIX,NEVAB,
     +     ICASE,NCASE,ITEMP,IPROB,NPROB
      COMMON/LGDATA/COORD(200,2),PROPS(10,
     +     4),PRESC(40,3),ASDIS(600),ELOAD
     +     (50,24),NOFIX(40),
     +     IFPRE(40,3),LNODS(50,8),MATNO
     +     (50)
      COMMON/WORK/ELCOD(2,8),SHAPE(8),
     +     DERIV(2,8),DMATX(5,5),CARTD
     +     (2,8),DBMAT(5,24),BMATX(5,
     +     24),SMATX(5,24,4),POSGP(2),
     +     WEIGP(2),GPCOD(2,4),NEROR(24)
C
      DO 10 ISTRE=1,NSTRE
      DO 10 IEVAB=1,NEVAB
         DBMAT(ISTRE,IEVAB)=0.0
      DO 10 JSTRE=1,NSTRE
         DBMAT(ISTRE,IEVAB)=DBMAT(ISTRE,IEVAB)+
     +   DMATX(ISTRE,JSTRE)*BMATX(JSTRE,IEVAB)
 10   CONTINUE
      RETURN
      END
C*************************************************
      SUBROUTINE STREPB
C
C*** CALCULATES STRESS RESULTANTS AT GAUSS
C    POINTS FOR PLATE BENDING ELEMENT
C
      implicit double precision (a-h,o-z)
      save
c
```

```
      DIMENSION ELDIS(3,8),STRSG(5)
C
      COMMON/CONTRO/NPOIN,NELEM,NNODE,NDOFN,NDIME,
     +     NSTRE,NTYPE,NGAUS,NPROP,NMATS,NVFIX,NEVAB,
     +     ICASE,NCASE,ITEMP,IPROB,NPROB
      COMMON/LGDATA/COORD(200,2),PROPS(10,
     +     4),PRESC(40,3),ASDIS(600),ELOAD
     +     (50,24),NOFIX(40),
     +     IFPRE(40,3),LNODS(50,8),MATNO
     +     (50)
      COMMON/WORK/ELCOD(2,8),SHAPE(8),
     +     DERIV(2,8),DMATX(5,5),CARTD
     +     (2,8),DBMAT(5,24),BMATX(5,
     +     24),SMATX(5,24,4),POSGP(2),
     +     WEIGP(2),GPCOD(2,4),NEROR(24)
C
      WRITE(6,900)
      WRITE(6,905)
C
C*** LOOP OVER EACH ELEMENT
C
      DO 40 IELEM=1,NELEM
C
C*** READ THE STRESS MATRIX, SAMPLING POINT
C    COORDINATES FOR THE ELEMENT
C
      READ(3) SMATX,GPCOD
      WRITE(6,910) IELEM
C
C*** IDENTIFY THE DISPLACEMENTS OF THE
C    ELEMENT NODAL POINTS
C
      DO 10 INODE=1,NNODE
      LNODE=LNODS(IELEM,INODE)
      NPOSN=(LNODE-1)*NDOFN
      DO 10 IDOFN=1,NDOFN
      NPOSN=NPOSN+1
      ELDIS(IDOFN,INODE)=ASDIS(NPOSN)
   10 CONTINUE
      kgasp=0
C
C*** ENTER LOOPS OVER EACH SAMPLING POINT
C
      DO 30 IGAUS=1,NGAUS
      DO 30 JGAUS=1,NGAUS
      KGASP=KGASP+1
      DO 20 ISTRE=1,NSTRE
      STRSG(ISTRE)=0.0
      KGASH=0
C
C*** COMPUTE THE STRESS RESULTANTS
C
      DO 20 INODE=1,NNODE
      DO 20 IDOFN=1,NDOFN
      KGASH=KGASH+1
      STRSG(ISTRE)=STRSG(ISTRE)+
```

```
     .SMATX(ISTRE,KGASH,KGASP)*ELDIS(IDOFN,INODE)
   20 CONTINUE
C
C*** OUTPUT THE STRESS RESULTANTS
C
      WRITE(6,915) KGASP,
     .(GPCOD(IDIME,KGASP),IDIME=1,NDIME),
     .(STRSG(ISTRE),ISTRE=1,NSTRE)
   30 CONTINUE
   40 CONTINUE
  900 FORMAT(/,10X,8HSTRESSES,/)
  905 FORMAT(1H0,4HG.P.,2X,8HX=COORD.,2X,
     .8HY=COORD.,3X,8HX-MOMENT,4X,8HY-MOMENT,
     .3X,9HXY-MOMENT,2X,10HXZ-S.FORCE,2X,
     .10HYZ-S.FORCE)
  910 FORMAT(/,5X,12HELEMENT NO.=,I5)
  915 FORMAT(I5,2F10.4,5E12.5)
      RETURN
      END
C*************************************************
      SUBROUTINE LOADPB(j)
C
C*** CALCULATE NODAL FORCES FOR PLATE ELEMENT
C
      implicit double precision (a-h,o-z)
      save
c
      include 'params.f'
      dimension parent(indmax,nparmax)
      dimension iparent(indmax,nchrmax)
      dimension moment(indmax,nparmax,2)
c
c      DIMENSION TITLE(12),POINT(3)
C
      COMMON/CONTRO/NPOIN,NELEM,NNODE,NDOFN,NDIME,
     +     NSTRE,NTYPE,NGAUS,NPROP,NMATS,NVFIX,NEVAB,
     +     ICASE,NCASE,ITEMP,IPROB,NPROB
      COMMON/LGDATA/COORD(200,2),PROPS(10,
     +     4),PRESC(40,3),ASDIS(600),ELOAD
     +     (50,24),NOFIX(40),
     +     IFPRE(40,3),LNODS(50,8),MATNO
     +     (50)
      COMMON/WORK/ELCOD(2,8),SHAPE(8),
     +     DERIV(2,8),DMATX(5,5),CARTD
     +     (2,8),DBMAT(5,24),BMATX(5,
     +     24),SMATX(5,24,4),POSGP(2),
     +     WEIGP(2),GPCOD(2,4),NEROR(24)
C
      common / ga2   / nparam,nchrome
      common / ga3  / parent,iparent
      common / load / unimoment(20,2),tarshape(200)
      common / HeatLine / itnl,coef(25,24,20,2)
c
c
      DO 10 IELEM=1,NELEM
      DO 10 IEVAB=1,NEVAB
```

```
      ELOAD(IELEM,IEVAB)=0.0
 10   CONTINUE
c      READ(5,900) TITLE
c 900  FORMAT(18A4)
c      WRITE(6,905) TITLE,ICASE
c 905  FORMAT(1H ,18A4,3X,12H LOAD CASE =,I5)
c      write(6,1000) icase
c 1000 format(12h load case =,I3)
C
C*** READ DATA CONTROLLING LOADING
C    TYPES TO BE INPUT
C
c      READ(5,910) IPLOD
c 910  FORMAT(I5)
c      WRITE(6,915) IPLOD
c 915  FORMAT(9h IPLOD = ,I5)
C
C*** READ NODAL POINT LOADS
C
c      IF(IPLOD.EQ.0) GO TO 60
c 20   READ(5,920)
c     .LODPT,(POINT(IDOFN),IDOFN=1,NDOFN)
c      WRITE(6,925)
c     .LODPT,(POINT(IDOFN),IDOFN=1,NDOFN)
c 920  FORMAT(I5,3F15.5)
c 925  FORMAT(I5,3F15.5)
C
C*** ASSOCIATE THE NODAL POINT LOADS
C    WITH AN ELEMENT
C
c      DO 30 IELEM=1,NELEM
c      DO 30 INODE=1,NNODE
c      NLOCA=LNODS(IELEM,INODE)
c      IF(LODPT.EQ.NLOCA) GO TO 40
c 30   CONTINUE
c 40   DO 50 IDOFN=1,NDOFN
c      NGASH=(INODE-1)*NDOFN+IDOFN
c      ELOAD(IELEM,NGASH)=POINT(IDOFN)
c
c      write(6,40) j
c 40   format(/,'output j from loadpb = ',i3)
c
c      write(6,1) unimoment
c 1    format(/,'output from loadpb   unimoment = ',f15.5)
c
      do 200 icount=1,nparam
         do 200 isign=1,2
c         write(6,20) icount,parent(j,icount),unimoment(icount,isign)
c 20      format(/,'output from loadpb parent(j',i3,') = ',f10.5,
c     +          ' unimoment=',f20.5)
         moment(j,icount,isign)=unimoment(icount,isign)*
     +                          (parent(j,icount)-2.0)
c         write(6,4) j,icount,isign,moment(j,icount,isign)
c 4       format('j=',i2,' icount=',i2,' isign=',i2,' moment=',
c     +          f20.5)
 200  continue
```

```
c       write(6,30) (parent(j,i),i=1,6)
c 30    format(/,'output from loadpb parents =',10f15.3)
c
c start of revision on 10/08/96
      do 2 ielem=1,nelem
         do 2 ngash=1,nevab
            do 2 inl=1,nparmax
               do 2 isign=1,2
         eload(ielem,ngash)=eload(ielem,ngash)+
     +    moment(j,inl,isign)*coef(ielem,ngash,inl,isign)
c         write(6,3) ielem,ngash,eload(ielem,ngash)
c 3       format('ielem=',i3,' ngash=',i3,' eload(ielem,ngash)='
c     +          f15.5)
 2       continue
c end of revision on 10/08/96
c
c 50    CONTINUE
c       IF(LODPT.NE.NPOIN) GO TO 20
 60    CONTINUE
C
C*** LOOP OVER EACH ELEMENT
C
C       write(6,1001) (nofix(i),i=1,nvfix)
C 1001 format(30h in LOADPB before loop nofix =,15i4)
      DO 110 IELEM=1,NELEM
      LPROP=MATNO(IELEM)
      UDLOD=PROPS(LPROP,4)
      IF(UDLOD.EQ.0.0) GO TO 110
C       write(6,1007) ielem,lprop,udlod,(nofix(i),i=1,nvfix)
C 1007 format(8h ielem =,i3,3x,8h lprop =,i3,3x,8h udlod =,
C     +i3,3x,8h nofix =,15i4/(20x,15i4))
C
C*** EVALUATE THE COORDINATES OF THE
C    ELEMENT NODAL POINTS
      DO 70 INODE=1,NNODE
      LNODE=LNODS(IELEM,INODE)
      DO 70 IDIME=1,NDIME
      ELCOD(IDIME,INODE)=COORD(LNODE,IDIME)
C       write(6,1008)inode,idime,lnode,elcod(idime,inode)
C 1008 format(8h inode =,i4,3x,8h idime =,i4,
C     +       8h lnode =,i4,3x,8h elcod =,f10.5)
C       write(6,1009)(nofix(i),i=1,nvfix)
C 1009 format(8h nofix =,40i3)
 70    CONTINUE
      DO 80 IEVAB=1,NEVAB
      ELOAD(IELEM,IEVAB)=0.0
C       write(6,1006) ievab,(nofix(i),i=1,nvfix)
C 1006 format(8h ievab =,i3,3x,8h nofix =,40i4)
 80    CONTINUE
      KGASP=0
C
C*** ENTER LOOPS FOR NUMERICAL INTEGRATION
C
      DO 100 IGAUS=1,NGAUS
      EXISP=POSGP(IGAUS)
      DO 100 JGAUS=1,NGAUS
```

```
      ETASP=POSGP(JGAUS)
      KGASP=KGASP+1
C
C*** EVALUATE THE SHAPE FUNCTIONS AT THE
C    SAMPLING POINTS AND ELEMENTAL AREA
C
C
      CALL SFR2(EXISP,ETASP)
      CALL JACOB2(IELEM,DJACB,KGASP)
      DAREA=DJACB*WEIGP(IGAUS)*WEIGP(JGAUS)
C
C*** CALCULATE LOADS AND ASSOCIATE WITH
C    ELEMENT NODAL POINTS
      DO 90 INODE=1,NNODE
      NPOSN=(INODE-1)*NDOFN+1
      ELOAD(IELEM,NPOSN)=ELOAD(IELEM,NPOSN)+
     .SHAPE(INODE)*UDLOD*DAREA
C      write(6,1003)ielem,inode,nposn,eload(ielem,nposn)
C 1003 format(8h ielem =,i3,3x,8h inode =,i3,3x,
C     +8h nposn =,i4,3x,8h eload =,f13.5)
C      write(6,1004) (nofix(i),i=1,nvfix)
C 1004 format(30h in LOADPB inside loop nofix =,15i4)
 90   CONTINUE
C      write(6,1005) igaus,jgaus,(nofix(i),i=1,nvfix)
C 1005 format(30h in LOADPB inside loop igaus =,i3,3x,
C     +8h jgaus =,i3,3x,8h nofix =,15i4/(50x,10i4))
 100  CONTINUE
C
C      write(6,1002) (nofix(i),i=1,nvfix)
C 1002 format(26h in LOADPB in loop nofix =,15i4)
C
 110  CONTINUE
c      WRITE(6,930)
c 930  FORMAT(1H0,5X,
c     .36H TOTAL NODAL FORCES FOR EACH ELEMENT)
      DO 120 IELEM=1,NELEM
c      WRITE(6,935) IELEM,
c     .(ELOAD(IELEM,IEVAB),IEVAB=1,NEVAB)
 120  CONTINUE
c 935  FORMAT(1X,I4,5X,8E12.4/(10X,8E12.4)/
c     .(10X,8E12.4))
      RETURN
      END
C*********************************************
      SUBROUTINE FRONT
C
      implicit double precision (a-h,o-z)
      save
c
      DIMENSION FIXED(600),EQUAT(60),VECRV(600),
     .GLOAD(60),GSTIF(1830),ESTIF(24,24),
     .IFFIX(600),NACVA(60),LOCEL(24),NDEST(24)
C
      COMMON/CONTRO/NPOIN,NELEM,NNODE,NDOFN,NDIME,
     +     NSTRE,NTYPE,NGAUS,NPROP,NMATS,NVFIX,NEVAB,
     +     ICASE,NCASE,ITEMP,IPROB,NPROB
```

```
      COMMON/LGDATA/COORD(200,2),PROPS(10,
     +     4),PRESC(40,3),ASDIS(600),ELOAD
     +     (50,24),NOFIX(40),
     +     IFPRE(40,3),LNODS(50,8),MATNO
     +     (50)
      COMMON/WORK/ELCOD(2,8),SHAPE(8),
     +     DERIV(2,8),DMATX(5,5),CARTD
     +     (2,8),DBMAT(5,24),BMATX(5,
     +     24),SMATX(5,24,4),POSGP(2),
     +     WEIGP(2),GPCOD(2,4),NEROR(24)
C
      NFUNC(I,J)=(J*J-J)/2+I
      MFRON=60
      MSTIF=1830
C
C*** INTERPRET FIXITY DATA IN VECTOR FORM
C
      NTOTV=NPOIN*NDOFN
      DO 100 ITOTV=1,NTOTV
      IFFIX(ITOTV)=0
C      write(6,1001)itotv,iffix(itotv)
C 1001 format(8h itotv =,i4,3x,8h iffix =,i4)
 100  FIXED(ITOTV)=0.0
      DO 110 IVFIX=1,NVFIX
C      write(6,1114) ivfix,nofix(ivfix)
C 1114 format(8h ivfix =,i4,3x,8h nofix =,i4)
      NLOCA=(NOFIX(IVFIX)-1)*NDOFN
      DO 110 IDOFN=1,NDOFN
      NGASH=NLOCA+IDOFN
      IFFIX(NGASH)=IFPRE(IVFIX,IDOFN)
      FIXED(NGASH)=PRESC(IVFIX,IDOFN)
C      write(6,1113) ivfix,ngash,iffix(ngash),fixed(ngash)
C 1113 format(8h ivfix =,i4,3x,8h ngash =,i4,3x,8h iffix =,
C     +       i4,3x,8h fixed =,f10.5)
 110  continue
C
C*** CHANGE THE SIGN OF THE LAST APPEARANCE
C    OF EACH NODE
C
      DO 140 IPOIN=1,NPOIN
      KLAST=0
      DO 130 IELEM=1,NELEM
      DO 120 INODE=1,NNODE
         IF(LNODS(IELEM,INODE).NE.IPOIN) GO TO 120
         KLAST=IELEM
         NLAST=INODE
 120     CONTINUE
 130     CONTINUE
         IF(KLAST.NE.0) LNODS(KLAST,NLAST)=-IPOIN
 140     CONTINUE
C
C*** START BY INITIALIZING EVERYTHING THAT
C    MATTERS TO ZERO
C
      DO 150 ISTIF=1,MSTIF
 150     GSTIF(ISTIF)=0.0
```

```
      DO 160 IFRON=1,MFRON
         GLOAD(IFRON)=0.0
         EQUAT(IFRON)=0.0
         VECRV(IFRON)=0.0
 160  NACVA(IFRON)=0
C
C*** AND PREPARE FOR DISC READING AND WRITING
C    OPERATIONS
C
      REWIND 1
      REWIND 2
      REWIND 3
      REWIND 4
C
C*** ENTER MAIN ELEMENT ASSEMBLY-REDUCTION LOOP
C
      NFRON=0
      KELVA=0
      DO 380 IELEM=1,NELEM
      KEVAB=0
      READ(1) ESTIF
      DO 170 INODE=1,NNODE
      DO 170 IDOFN=1,NDOFN
      NPOSI=(INODE-1)*NDOFN+IDOFN
      LOCNO=LNODS(IELEM,INODE)
      IF(LOCNO.GT.0) LOCEL(NPOSI)=(LOCNO-1)*
     .NDOFN+IDOFN
      IF(LOCNO.LT.0) LOCEL(NPOSI)=(LOCNO+1)*
     .NDOFN-IDOFN
 170  CONTINUE
C      write(6,1100) (locel(i),i=1,nevab)
C 1100 format(8h locel =,10i4/(10i4)/(10i4))
C
C*** START BY LOOKING FOR EXISTING DESTINATIONS
C
      DO 210 IEVAB=1,NEVAB
         NIKNO=IABS(LOCEL(IEVAB))
C      write(6,1105) ievab,locel(ievab),nikno
C 1105 format(8h ievab =,i3,15h locel(ievab) =,i3,8h nikno =,i3)
         KEXIS=0
         DO 180 IFRON=1,NFRON
            IF(NIKNO.NE.NACVA(IFRON)) GO TO 180
            KEVAB=KEVAB+1
            KEXIS=1
            NDEST(KEVAB)=IFRON
 180     CONTINUE
      IF(KEXIS.NE.0) GO TO 210
C
C*** WE NOW SEEK NEW EMPTY PLACES FOR
C    DESTINATION VECTOR
C
      DO 190 IFRON=1,MFRON
         IF(NACVA(IFRON).NE.0) GO TO 190
         NACVA(IFRON)=NIKNO
C      write(6,1101) ifron,nacva(ifron)
C 1101 format(8h ifron =,i3,15h nacva(ifron) =,i3)
```

```
         KEVAB=KEVAB+1
         NDEST(KEVAB)=IFRON
         GO TO 200
 190     CONTINUE
C
C*** THE NEW PLACES MAY DEMAND AN INCREASE
C    IN CURRENT FRONTWIDTH
C
 200  IF(NDEST(KEVAB).GT.NFRON) NFRON=NDEST(KEVAB)
C      write(6,1102) ievab,(nacva(i),i=1,mfron)
C 1102 format(8h ievab =,i3,8h nacva =,15i4/(15i4)/(15i4))
 210  CONTINUE
C
C*** ASSEMBLE ELEMENT LOADS
C
      DO 240 IEVAB=1,NEVAB
         IDEST=NDEST(IEVAB)
         GLOAD(IDEST)=GLOAD(IDEST)+ELOAD(IELEM,IEVAB)
C
C*** ASSEMBLE THE ELEMENT STIFFNESSES
C    - BUT NOT IN RESOLUTION
C
         IF(ICASE.GT.1) GO TO 230
         DO 220 JEVAB=1,IEVAB
            JDEST=NDEST(JEVAB)
            NGASH=NFUNC(IDEST,JDEST)
            NGISH=NFUNC(JDEST,IDEST)
            IF(JDEST.GE.IDEST) GSTIF(NGASH)=
     .      GSTIF(NGASH)+ESTIF(IEVAB,JEVAB)
            IF(JDEST.LT.IDEST) GSTIF(NGISH)=
     .      GSTIF(NGISH)+ESTIF(IEVAB,JEVAB)
 220        CONTINUE
C      write(6,1103) (gstif(i),i=1,mstif)
C 1103 format(8h gstif = ,10f10.6/(10f10.6)/(10f10.6))
 230        CONTINUE
 240        CONTINUE
C       write(6,1106) (gload(i),i=1,mfron)
C 1106  format(8h gload =,10f10.5/(10f10.5)/(10f10.5))
C
C*** RE-EXAMINE EACH ELEMENT NODE, TO
C    ENQUIRE WHICH CAN BE ELIMINATED
C
      DO 370 IEVAB=1,NEVAB
         NIKNO=-LOCEL(IEVAB)
         IF(NIKNO.LE.0) GO TO 370
C
C*** FIND POSITIONS OF VARIABLES READY
C    FOR ELIMINATION
C
      DO 350 IFRON=1,NFRON
      IF(NACVA(IFRON).NE.NIKNO) GO TO 350
C
C*** EXTRACT THE COEFFICIENTS OF THE
C    NEW EQUATION FOR ELIMINATION
C
C
```

```
      IF(ICASE.GT.1) GO TO 260
      DO 250 JFRON=1,MFRON
         IF(IFRON.LT.JFRON) NLOCA=NFUNC(IFRON,JFRON)
         IF(IFRON.GE.JFRON) NLOCA=NFUNC(JFRON,IFRON)
         EQUAT(JFRON)=GSTIF(NLOCA)
 250  GSTIF(NLOCA)=0.0
C      write(6,1107) (equat(i),i=1,mfron)
C 1107 format(8h equat =,10f10.5/(10f10.5)/(10f10.5))
 260  CONTINUE
C
C*** AND EXTRACT THE CORRESPONDING RIGHT
C    HAND SIDES
C
      EQRHS=GLOAD(IFRON)
      GLOAD(IFRON)=0.0
      KELVA=KELVA+1
C
C*** WRITE EQUATIONS TO DISC OR TAPE
C
      IF(ICASE.GT.1) GO TO 270
      WRITE(2) EQUAT,EQRHS,IFRON,NIKNO
      GO TO 280
 270  WRITE(4) EQRHS
      READ(2) EQUAT,DUMMY,IDUMM,NIKNO
 280  CONTINUE
C
C*** DEAL WITH PIVOT
C
      PIVOT=EQUAT(IFRON)
C      write(6,1000) ielem,ievab,ifron,pivot,(equat(i),i=1,mfron)
C 1000 format(/3i3,5x,f12.6,8f10.5/(26x,8f10.5)/
C     +(26x,8f10.5))
      EQUAT(IFRON)=0.0
C
C*** ENQUIRE WHETHER PRFSFNT VARIABLE IS
C    FREE OR PRESCRIBED
C
C      write(6,1112) nikno,iffix(nikno)
C 1112 format(8h nikno =,i4,3x,8h iffix =,i3)
      IF(IFFIX(NIKNO).EQ.0) GO TO 300
C
C*** DEAL WITH A PRESCRIBED DEFLECTION
C
      DO 290 JFRON=1,NFRON
 290     GLOAD(JFRON)=GLOAD(JFRON)-FIXED(NIKNO)*
     .EQUAT(JFRON)
      GO TO 340
C
C*** ELEMINATE A FREE VARIABLE - DEAL WITH
C    THE RIGHT HAND SIDE FIRST
C
 300  DO 330 JFRON=1,NFRON
         GLOAD(JFRON)=GLOAD(JFRON)-EQUAT(JFRON)*
     .EQRHS/PIVOT
C
C*** NOW DEAL WITH THE COEFFICIENTS IN CORE
```

```
C
         IF(ICASE.GT.1) GO TO 320
         IF(EQUAT(JFRON).EQ.0.0) GO TO 330
         NLOCA=NFUNC(0,JFRON)
         DO 310 LFRON=1,JFRON
            NGASH=LFRON+NLOCA
 310     GSTIF(NGASH)=GSTIF(NGASH)-EQUAT(JFRON)*
     .EQUAT(LFRON)/PIVOT
 320  CONTINUE
 330  CONTINUE
 340  EQUAT(IFRON)=PIVOT
C
C*** RECORD THE NEW VACANT SPACE, AND REDUCE
C    FRONTWIDTH IF POSSIBLE
C
      NACVA(IFRON)=0
      GO TO 360
C
C*** COMPLETE THE ELEMENT LOOP IN THE FORWARD
C    ELIMINATION
C
 350  CONTINUE
 360  IF(NACVA(NFRON).NE.0) GO TO 370
      NFRON=NFRON-1
      IF(NFRON.GT.0) GO TO 360
 370  CONTINUE
 380  CONTINUE
C
C*** ENTER BACK-SUBSTITUTION PHASE, LOOP
C    BACKWARDS THROUGH VARIABLES
C
       DO 410 IELVA=1,KELVA
C
C*** READ A NEW EQUATION
C
          BACKSPACE 2
          READ(2) EQUAT,EQRHS,IFRON,NIKNO
          BACKSPACE 2
          IF(ICASE.EQ.1) GO TO 390
          BACKSPACE 4
          READ(4) EQRHS
          BACKSPACE 4
 390      CONTINUE
C
C*** PREPARE TO BACK-SUBSTITUTE FROM THE
C    CURRENT EQUATION
C
          PIVOT=EQUAT(IFRON)
C      write(6,1003)nikno,iffix(nikno)
C 1003 format(8h nikno =,i4,3x,8h iffix =,i4)
          IF(IFFIX(NIKNO).EQ.1) VECRV(IFRON)=
     .FIXED(NIKNO)
          IF(IFFIX(NIKNO).EQ.0) EQUAT(IFRON)=0.0
C
C*** BACK-SUBSTITUTE IN THE CURRENT EQUATION
C
```

```
          DO 400 JFRON=1,MFRON
 400      EQRHS=EQRHS-VECRV(JFRON)*EQUAT(JFRON)
C
C*** PUT THE FINAL VALUES WHERE THEY BELONG
C
          IF(IFFIX(NIKNO).EQ.0) VECRV(IFRON)=
     .EQRHS/PIVOT
          IF(IFFIX(NIKNO).EQ.1) FIXED(NIKNO)=-EQRHS
          ASDIS(NIKNO)=VECRV(IFRON)
C      write(6,1111) nikno,iffix(nikno),ifron,vecrv(ifron),fixed(nikno),
C     +              asdis(nikno)
C 1111 format(8h nikno =,i4,3x,8h iffix =,i4,3x,8h ifron =,i4,3x,
C     +       8h vecrv =,f13.5,3x,8h fixed =,f10.5,3x,8h asdis =,f13.5)
 410  CONTINUE
cc      WRITE(6,900)
cc 900  FORMAT(1H0,5X,13HDISPLACEMENTS)
cc      IF(NDOFN.NE.2) GO TO 430
cc      IF(NDIME.NE.1) GO TO 420
cc      WRITE(6,905)
cc 905  FORMAT(1H0,5X,4HNODE,6X,5HDISP.,7X,
cc     .8HROTATION)
cc      GO TO 440
cc 420  WRITE(6,910)
cc 910  FORMAT(1H0,5X,4HNODE,5X,7HX-DISP.,
cc     .7X,7HY-DISP.)
cc      GO TO 440
cc 430  WRITE(6,915)
cc 915  FORMAT(1H0,5X,4HNODE,6X,5HDISP.,8X,
cc     .7HXZ-ROT.,7X,7HYZ-ROT.)
cc 440  CONTINUE
cc      DO 450 IPOIN=1,NPOIN
cc         NGASH=IPOIN*NDOFN
cc         NGISH=NGASH-NDOFN+1
cc 450     WRITE(6,920) IPOIN,(ASDIS(IGASH),IGASH=
cc     .NGISH,NGASH)
cc 920  FORMAT(I10,3E14.6)
cc      WRITE(6,925)
cc 925  FORMAT(1H0,5X,9HREACTIONS)
cc      IF(NDOFN.NE.2) GO TO 470
cc      IF(NDIME.NE.1) GO TO 460
cc      WRITE(6,930)
cc 930  FORMAT(1H0,5X,4HNODE,6X,5HFORCE,8X,6HMOMENT)
cc      GO TO 480
cc 460  WRITE(6,935)
cc 935  FORMAT(1H0,5X,4HNODE,5X,7HX-FORCE,7X,
cc     .7HY-FORCE)
cc      GO TO 480
cc 470  WRITE(6,940)
cc 940  FORMAT(1H0,5X,4HNODE,6X,5HFORCE,6X,
cc     .9HXZ-MOMENT,5X,9HYZ-MOMENT)
cc 480  CONTINUE
cc      DO 510 IPOIN=1,NPOIN
cc         NLOCA=(IPOIN-1)*NDOFN
cc         DO 490 IDOFN=1,NDOFN
cc            NGUSH=NLOCA+IDOFN
C      write(6,1000)ipoin,nloca,idofn,ngush,iffix(ngush)
```

```
C 1000 format(8h ipoin =,i3,3x,8h nloca =,i5,3x,8h idofn =,
C     +       i5,3x,8h ngush =,i5,3x,8h iffix =,i5)
cc            IF(IFFIX(NGUSH).GT.0) GO TO 500
cc 490      CONTINUE
cc       GO TO 510
cc 500   NGASH=NLOCA+NDOFN
cc       NGISH=NLOCA+1
cc       WRITE(6,945) IPOIN,(FIXED(IGASH),IGASH=
cc      .NGISH,NGASH)
cc 510   CONTINUE
cc 945   FORMAT(I10,3E14.6)
C
C*** POST FRONT-RESET ALL ELEMENT CONNECTION
C    NUMBERS TO POSITIVE VALUES FOR SUBSEQUENT
C    USE IN STRESS CALCULATION
C
      DO 520 IELEM=1,NELEM
         DO 520 INODE=1,NNODE
 520  LNODS(IELEM,INODE)=IABS(LNODS(IELEM,INODE))
      RETURN
      END
C*****************************************************
      SUBROUTINE CHECK1
C
C*** TO CRITICIZE THE DATA CONTROL CARD AND
C    PRINT ANY DIAGNOSTICS
C
      implicit double precision (a-h,o-z)
      save
c
      COMMON/CONTRO/NPOIN,NELEM,NNODE,NDOFN,NDIME,
     +     NSTRE,NTYPE,NGAUS,NPROP,NMATS,NVFIX,NEVAB,
     +     ICASE,NCASE,ITEMP,IPROB,NPROB
      COMMON/LGDATA/COORD(200,2),PROPS(10,
     +     4),PRESC(40,3),ASDIS(600),ELOAD
     +     (50,24),NOFIX(40),
     +     IFPRE(40,3),LNODS(50,8),MATNO
     +     (50)
      COMMON/WORK/ELCOD(2,8),SHAPE(8),
     +     DERIV(2,8),DMATX(5,5),CARTD
     +     (2,8),DBMAT(5,24),BMATX(5,
     +     24),SMATX(5,24,4),POSGP(2),
     +     WEIGP(2),GPCOD(2,4),NEROR(24)
C
      DO 10 IEROR=1,24
 10   NEROR(IEROR)=0
C
C*** CREATE THE DIAGNOSTIC MESSAGES
C
      IF(NPOIN.LE.0) NEROR(1)=1
      IF(NELEM*NNODE.LT.NPOIN) NEROR(2)=1
      IF(NVFIX.LT.1.OR.NVFIX.GT.NPOIN) NEROR(3)=1
      IF(NCASE.LE.0) NEROR(4)=1
      IF(NTYPE.LT.0.OR.NTYPE.GT.2) NEROR(5)=1
      IF(NNODE.LT.3.OR.NNODE.GT.8) NEROR(6)=1
      IF(NDOFN.LT.2.OR.NDOFN.GT.3) NEROR(7)=1
```

```
      IF(NMATS.LE.0.OR.NMATS.GT.NELEM) NEROR(8)=1
      IF(NPROP.LT.3.OR.NPROP.GT.5) NEROR(9)=1
      IF(NGAUS.LT.2.OR.NGAUS.GT.3) NEROR(10)=1
      IF(NDIME.LT.1.OR.NDIME.GT.2) NEROR(11)=1
      IF(NSTRE.LT.2.OR.NSTRE.GT.5) NEROR(12)=1
C
C*** EITHER RETURN, OR ELSE PRINT THE ERRORS
C    DIAGNOSED
C
      KEROR=0
      DO 20 IEROR=1,12
         IF(NEROR(IEROR).EQ.0) GO TO 20
         KEROR=1
         WRITE(6,900) IEROR
 900     FORMAT(//25H *** DIAGNOSIS BY CHECK1,
     .6H EROR,I3)
 20   CONTINUE
      IF(KEROR.EQ.0) RETURN
C
C*** OTHERWISE ECHO ALL THE REMAINING DATA
C    WITHOUT FURTHER COMMENT
C
      CALL ECHO
      END
C*******************************************************
      SUBROUTINE ECHO
C
      implicit double precision (a-h,o-z)
      save
c
      DIMENSION NTITL(80)
C
      COMMON/CONTRO/NPOIN,NELEM,NNODE,NDOFN,NDIME,
     +     NSTRE,NTYPE,NGAUS,NPROP,NMATS,NVFIX,NEVAB,
     +     ICASE,NCASE,ITEMP,IPROB,NPROB
      COMMON/LGDATA/COORD(200,2),PROPS(10,
     +     4),PRESC(40,3),ASDIS(600),ELOAD
     +     (50,24),NOFIX(40),
     +     IFPRE(40,3),LNODS(50,8),MATNO
     +     (50)
      COMMON/WORK/ELCOD(2,8),SHAPE(8),
     +     DERIV(2,8),DMATX(5,5),CARTD
     +     (2,8),DBMAT(5,24),BMATX(5,
     +     24),SMATX(5,24,4),POSGP(2),
     +     WEIGP(2),GPCOD(2,4),NEROR(24)
C
      WRITE(6,900)
 900  FORMAT(//25H NOW FOLLOWS A LISTING OF,
     .25H POST-DISASTER DATA CARDS/)
 10   READ(5,905) NTITL
 905  FORMAT(80A1)
      WRITE(6,910) NTITL
 910  FORMAT(20X,80A1)
      GO TO 10
      END
C*******************************************************
```

```
      SUBROUTINE CHECK2
C
C*** TO CRITICIZE THE DATA FROM SUBROUTINE INPUT
C
      implicit double precision (a-h,o-z)
      save
c
      DIMENSION NDFRO(25)
C
      COMMON/CONTRO/NPOIN,NELEM,NNODE,NDOFN,NDIME,
     +     NSTRE,NTYPE,NGAUS,NPROP,NMATS,NVFIX,NEVAB,
     +     ICASE,NCASE,ITEMP,IPROB,NPROB
      COMMON/LGDATA/COORD(200,2),PROPS(10,
     +     4),PRESC(40,3),ASDIS(600),ELOAD
     +     (50,24),NOFIX(40),
     +     IFPRE(40,3),LNODS(50,8),MATNO
     +     (50)
      COMMON/WORK/ELCOD(2,8),SHAPE(8),
     +     DERIV(2,8),DMATX(5,5),CARTD
     +     (2,8),DBMAT(5,24),BMATX(5,
     +     24),SMATX(5,24,4),POSGP(2),
     +     WEIGP(2),GPCOD(2,4),NEROR(24)
C
      MFRON=60
C
C*** CHECK AGAINST TWO IDENTICAL NONZERO
C    NODAL COORDINATES
C
      DO 10 IELEM=1,NELEM
 10   NDFRO(IELEM)=0
      DO 40 IPOIN=2,NPOIN
         KPOIN=IPOIN-1
         DO 30 JPOIN=1,KPOIN
            DO 20 IDIME=1,NDIME
            IF(COORD(IPOIN,IDIME).NE.COORD(JPOIN,
     .IDIME)) GO TO 30
 20         CONTINUE
         NEROR(13)=NEROR(13)+1
 30      CONTINUE
 40   CONTINUE
C
C*** CHECK THE LIST OF ELEMENT PROPERTY NUMBERS
C
      DO 50 IELEM=1,NELEM
 50      IF(MATNO(IELEM).LE.0.OR.MATNO(IELEM).GT.
     .NMATS) NEROR(14)=NEROR(14)+1
C
C*** CHECK FOR IMPOSSIBLE NODE NUMBERS
C
      DO 70 IELEM=1,NELEM
         DO 60 INODE=1,NNODE
            IF(LNODS(IELEM,INODE).EQ.0) NEROR(15)=
     .NEROR(15)+1
 60      IF(LNODS(IELEM,INODE).LT.0.OR.LNODS(IELEM,
     .INODE).GT.NPOIN) NEROR(16)=NEROR(16)+1
 70   CONTINUE
```

```
C
C*** CHECK FOR ANY REPETITION OF A NODE
C    NUMBER WITHIN AN ELEMENT
C
      DO 140 IPOIN=1,NPOIN
         KSTAR=0
         DO 100 IELEM=1,NELEM
            KZERO=0
            DO 90 INODE=1,NNODE
            IF(LNODS(IELEM,INODE).NE.IPOIN) GO TO 90
            KZERO=KZERO+1
            IF(KZERO.GT.1) NEROR(17)=NEROR(17)+1
C
C*** SEEK FIRST, LAST AND INTERMEDIATE
C    APPEATRANCES OF NODE IPOIN
C
            IF(KSTAR.NE.0) GO TO 80
            KSTAR=IELEM
C
C*** CALCULATE INCREASE OR DECREASE IN
C    FRONTWIDTH AT EACH ELEMENT STAGE
C
            NDFRO(IELEM)=NDFRO(IELEM)+NDOFN
 80         CONTINUE
C
C*** AND CHANGE THE SIGN OF THE LAST
C    APPEARANCE OF EACH NODE
C
            KLAST=IELEM
            NLAST=INODE
 90      CONTINUE
 100     CONTINUE
         IF(KSTAR.EQ.0) GO TO 110
         IF(KLAST.LT.NELEM) NDFRO(KLAST+1)=
     .NDFRO(KLAST+1)-NDOFN
         LNODS(KLAST,NLAST)=-IPOIN
         GO TO 140
C
C*** CHECH THAT COORDINATES FOR AN UNUSED
C    NODE HAVE NOT BEEN SPECIFIED
C
 110     WRITE(6,900) IPOIN
 900     FORMAT(/15H CHECH WHY NODE,I4,
     .14H NEVER APPEARS)
         NEROR(18)=NEROR(18)+1
         SIGMA=0.0
         DO 120 IDIME=1,NDIME
 120     SIGMA=SIGMA+ABS(COORD(IPOIN,IDIME))
         IF(SIGMA.NE.0.0) NEROR(19)=NEROR(19)+1
C
C*** CHECK THAT AN UNUSED NODE NUMBER IS NOT
C    A RESTRAINED NODE
C
         DO 130 IVFIX=1,NVFIX
 130     IF(NOFIX(IVFIX).EQ.IPOIN) NEROR(20)=
     .NEROR(20)+1
```

```
 140  CONTINUE
C
C*** CALCULATE THE LARGEST FRONTWIDTH
C
      NFRON=0
      KFRON=0
      DO 150 IELEM=1,NELEM
         NFRON=NFRON+NDFRO(IELEM)
 150  IF(NFRON.GT.KFRON) KFRON=NFRON
      WRITE(6,905) KFRON
 905  FORMAT(//28HMAX FRONTWIDTH ENCOUNTERED =,I5)
      IF(KFRON.GT.MFRON) NEROR(21)=1
C
C*** CONTINUE CHECKING THE DATA FOR THE
C    FIXED VALUES
C
      DO 170 IVFIX=1,NVFIX
         IF(NOFIX(IVFIX).LE.0.OR.NOFIX(IVFIX).
     .GT.NPOIN) NEROR(22)=NEROR(22)+1
         KOUNT=0
         DO 160 IDOFN=1,NDOFN
 160     IF(IFPRE(IVFIX,IDOFN).GT.0) KOUNT=1
         IF(KOUNT.EQ.0) NEROR(23)=NEROR(23)+1
         KVFIX=IVFIX-1
         DO 170 JVFIX=1,KVFIX
 170  IF(IVFIX.NE.1.AND.NOFIX(IVFIX).EQ.
     .NOFIX(JVFIX)) NEROR(24)=NEROR(24)+1
      KEROR=0
      DO 180 IEROR=13,24
         IF(NEROR(IEROR).EQ.0) GO TO 180
         KEROR=1
         WRITE(6,910) IEROR,NEROR(IEROR)
 910     FORMAT(//30H*** DIAGNOSIS BY CHECK2, ERROR,
     .I3,6X,18H ASSOCIATED NUMBER,I5)
 180  CONTINUE
      IF(KEROR.NE.0) GO TO 200
C
C*** RETURN ALL NODAL CONNECTION NUMBERS TO
C    POSITIVE VALUES
C
      DO 190 IELEM=1,NELEM
         DO 190 INODE=1,NNODE
 190  LNODS(IELEM,INODE)=IABS(LNODS(IELEM,INODE))
      RETURN
 200  CALL ECHO
      END
C*********************************************************
```

Appendix B

Final Shape Measurement of Laser Forming

B.1 Introduction

The experiments were conducted at the Applied Research Laboratory (ARL), Pennsylvania State University (PSU) under the instruction of Mr. Richard P. Martukanitz and Mr. Grant K. Sumnicht. Most experiments were one-dimensional laser forming experiments in which one heat process was put on in the middle of a square plate. Eight kinds of material specimens and two kinds of laser process apparatus were used for the experiments. Five series of experiments were conducted. Among these five series, final shapes were measured in experiment 2 from January 22 to February 16, 1996, experiment 4 from October 28 to November 2, 1996, and experiment 5 from December 16 to December 17, 1996.

B.2 Descriptions of Experiments

Specimens, laser process apparatus, process parameters, and experimental procedures are described in this section.

Table B.1: List of Specimens

Material	Thickness (mm)	Dimensions (mm)
HSLA-80	6.35	304.8 x 304.8
HY-80	7.02	304.8 x 304.8
Inconel 625	6.35	304.8 x 304.8
Inconel 718	6.55	304.8 x 304.8
A2219-o	6.35	304.8 x 304.8
A5052-o	6.38	304.8 x 304.8
Ti-6Al-4V	5.12	304.8 x 304.8
Ti-15Al-3V	1.8	304.8 x 304.8

B.2.1 Specimens

We used eight kinds of material; HSLA-80, HY-80, Inconel 625, Inconel 718, A2219-o, A5052-o, Ti-6Al-4V, and Ti-15Al-3V. All the specimens were squares in the same size of 304.8mm (12") × 304.8 mm(12"). The range of thickness was from 1.8mm (0.071") to 7.02mm (0.276").

B.2.2 Laser Process Apparatus

Process facilities consist of a laser oscillator, a laser beam transfer, and a laser head traverser. Two kinds of laser beams were used in the experiment: CO_2 laser and Nd:YAG laser. Each laser has the specific distribution of power at the exit of the process head. The distributions of laser beams were measured at PSU. Also, these two kinds of laser beams have different absorption rates for materials.

Generally speaking, we can obtain relatively high power output in CO_2 laser beam. This is the reason why it is the most popular kind of the laser in industrial field and our research team tried to focus on the investigation of its characteristics at the first stage. However, after the comparison of deformations produced by the two lasers, we realized that Nd:YAG laser is more efficient and versatile for practical uses, since the CO_2 laser has high absorption rate for most kinds of material and its beam cannot be transferred by fiber optics. Then our focus of the research gradually moved to the investigation of Nd:YAG laser characteristics.

CO_2 Laser

We used the United Technology Industrial Lasers Model 21. Its maximum output is 14kW. The laser beam is transferred by mirrors to a process table with a process head moved by a gantry-type traverser which is manufactured by Laser Articulating Robotics System (LARS). Going through the delivery system, a laser beam was focused by an f 10 reflective focus optics.

Nd:YAG Laser

We used a 3kW Nd:YAG laser beam oscillator manufactured by HOBART TAFA whose model number is 3000. A laser beam generated by the oscillator was transferred by an optical fiber tube to a heat process head at the site. A laser beam was focused at a process head by f 2 focus optics. Although the energy of the laser beam was dispersed in the fiber tube, the output of the oscillator was adjusted to keep the output on a specimen 2.6 kW constantly. The laser beam were delivered by an optical fiber to a process table and focused by lenses in a laser process head. The process head is traversed by a six-axis articulated robot manufactured by Yasukawa.

B.2.3 Heat Process Parameters

We have six parameters to define a heat process: laser power, spot size, inclination angle, process velocity, coating on specimen, and backing condition. Laser power, spot size, and process velocity are considered as the primary parameters which have significant effect on deflection. On the other hand, inclination angle and coating on specimen, and backing condition are considered as the secondary parameters which have no significant effect on the deflection of the specimens. (See Figure B-1)

Laser Power

A laser beam was radiated onto a specimen. Although the laser power is controlled at the oscillator, it is not suitable for experiments since there is energy loss at the delivery system of laser beam. Then we measured the laser power on the specimen

utilizing the calorimeter. All laser power shown in this report are the power on the specimens.

Stand-Off Distance and Spot Size

The spot size is an important parameter for laser processing since it represents the heat density of a laser beam. As the laser beam is focused by lenses, the spot size changes with the stand-off distance.

Process Velocity

The heat penetration is changed by the velocity of process.

Inclination Angle

A laser head has an inclination angle to protect the laser system from the reflection of the laser beam itself. Figure B-1 shows the definition of the inclination angle in a heat process.

Coating on Specimen

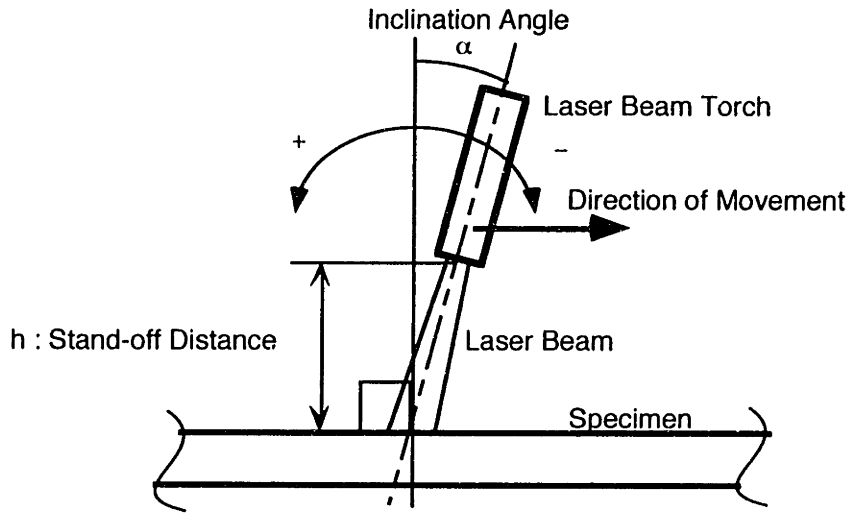
The coating on specimens is applied to reduce the reflection of a laser beam. Two types of coating were used in the experiments: graphite and stove black.

Backing Condition

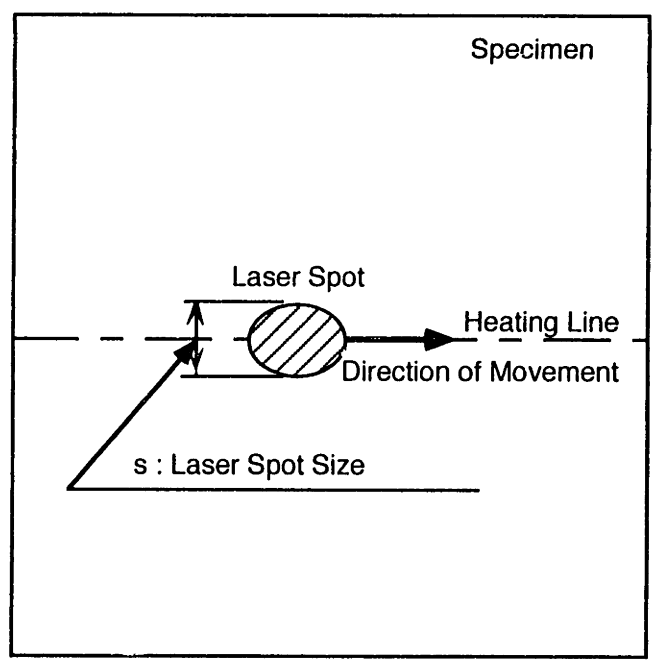
Basically as specimens were hanged on a support, they were cooled naturally by convection. Some specimens were put on an Aluminum plate to increase the cooling rate of the specimens.

B.2.4 Experimental Procedure

A laser beam traveled on a plate in a straight line in the middle of a plate from a edge through another edge. All heat process parameters were constant during each process.



(a) Inclination Angle (α) and Stand-off Distance (h) in the Side View of a Process



(b) Laser Spot Size (s) in the Top View of a Process

Figure B-1: Definitions of parameters of a heat process

Table B.2: Specifications of Three Dimensional Coordinator

Manufacturer	Brown & Sharpe
Model	MicroVal PFx
Repeatability	0.003mm
Volumetric Accuracy	0.010mm
Linear Accuracy	0.005mm
Resolution	0.001mm
Range XYZ	457 x 508 x 406 mm

B.3 Descriptions of Measurement

The plates were sent to Massachusetts Institute of Technology (MIT), Cambridge after the experiments and the post-process dimensions were determined with a Brown & Sharpe Micro Val Pfx coordinate measuring machine (CMM) owned by the Laboratory for Manufacturing and Productivity at the Department of Mechanical Engineering of MIT.

B.3.1 Measurement Equipment

The specifications of the coordinate measuring machine used for the measurement of the final shapes of the specimens are shown in Table B.2. The CMM has the function of automatic measuring. Defining the coordinate system for each specimen, the CMM automatically measures the displacement of the specimen from the base plane in z-axis. The appearance of the CMM is shown in Figure B-2.

B.3.2 Measurement Points and Angular Distortion

The specimens were carefully put on the stone base plate of the coordinate measuring machine with four wedges beneath the corners as supports with the top side of the process down. The orientation of the coordinate system was determined with the positions of four points on two edges of each plate. x-axis was defined as the axis perpendicular to the heating line and y-axis as the axis parallel to the heating line. After the setting of the orientation of the coordinate system, the measuring area was defined as 5.0mm $\leq x \leq$ 300.0mm and 5.0mm $\leq y \leq$ 300.0mm. Total 225

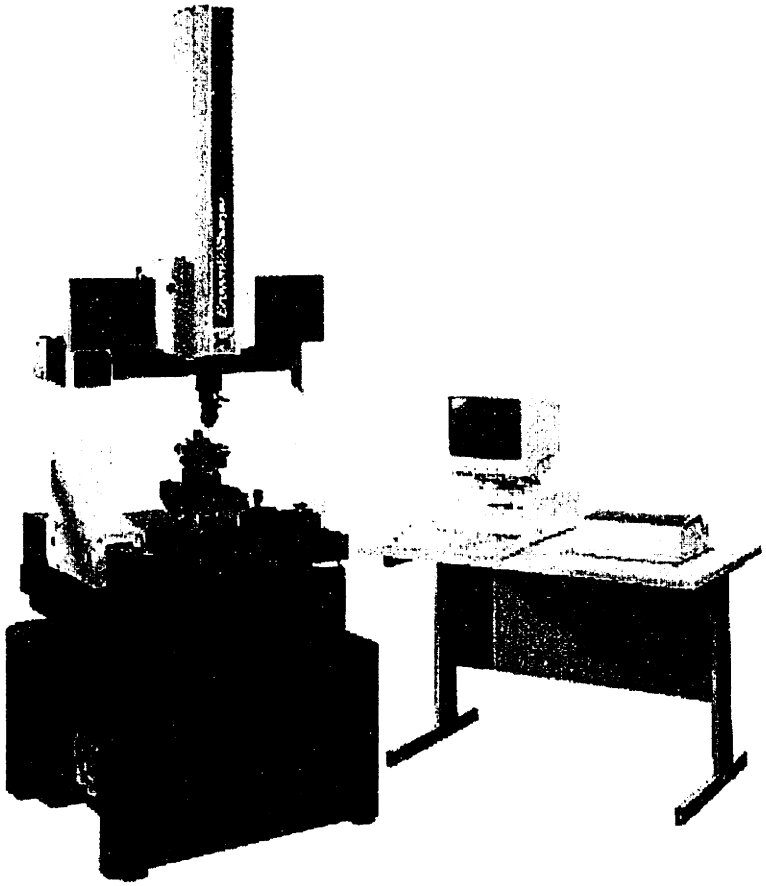


Figure B-2: Three dimensional coordinate measuring machine, MicroVal PFx manufactured by Brown & Sharpe

measurement points were determined as the intersections of fifteen measuring lines parallel to the x-axis at even intervals and fifteen measuring lines parallel to the y-axis at even intervals. The displacement from the base plane of the CMM to the bottom surface of the process was measured at each measurement point. (See Figure B-3)

The angular distortions were calculated from the z-axis displacements as results, assuming the curvatures caused by one-dimensional heat process are local and considering the profiles of specimens as folded lines. Total number of angular distortions calculated in a specimen was 15. The calculation of an angular distortion requires four displacements along a measuring line parallel to the x-axis. The positions of x=26.08mm, x=89.03mm, x=215.73mm, and x=278.93mm are selected to calculate the angular distortion. As the curvatures of the final shape were localized close to the heating line, the profiles of the specimen were expected to be straight lines between x=26.08mm and x=89.03mm and between x=215.73mm and x=278.93mm.

B.4 Results

The results of experiment 2, 4, and 5 were presented in this section.

B.4.1 Taguchi Experiments with CO_2 Laser (Experiment 2)

Process parameters that have the greatest influence on deflection by one-dimensional laser forming were evaluated through the use of a Taguchi experiment matrix for the CO_2 laser. Four parameters were selected for the independent parameters of the matrix: backing condition, coating type, spot size, and process velocity. On the other hand, material, thickness, and laser power were kept constant during this series of the experiments.

Table B.3 shows the process conditions of the experiments and Table B.4 shows the angular distortions. The final shapes of 1T and INC625-1 are shown in Figure B-4 and Figure B-5 respectively.

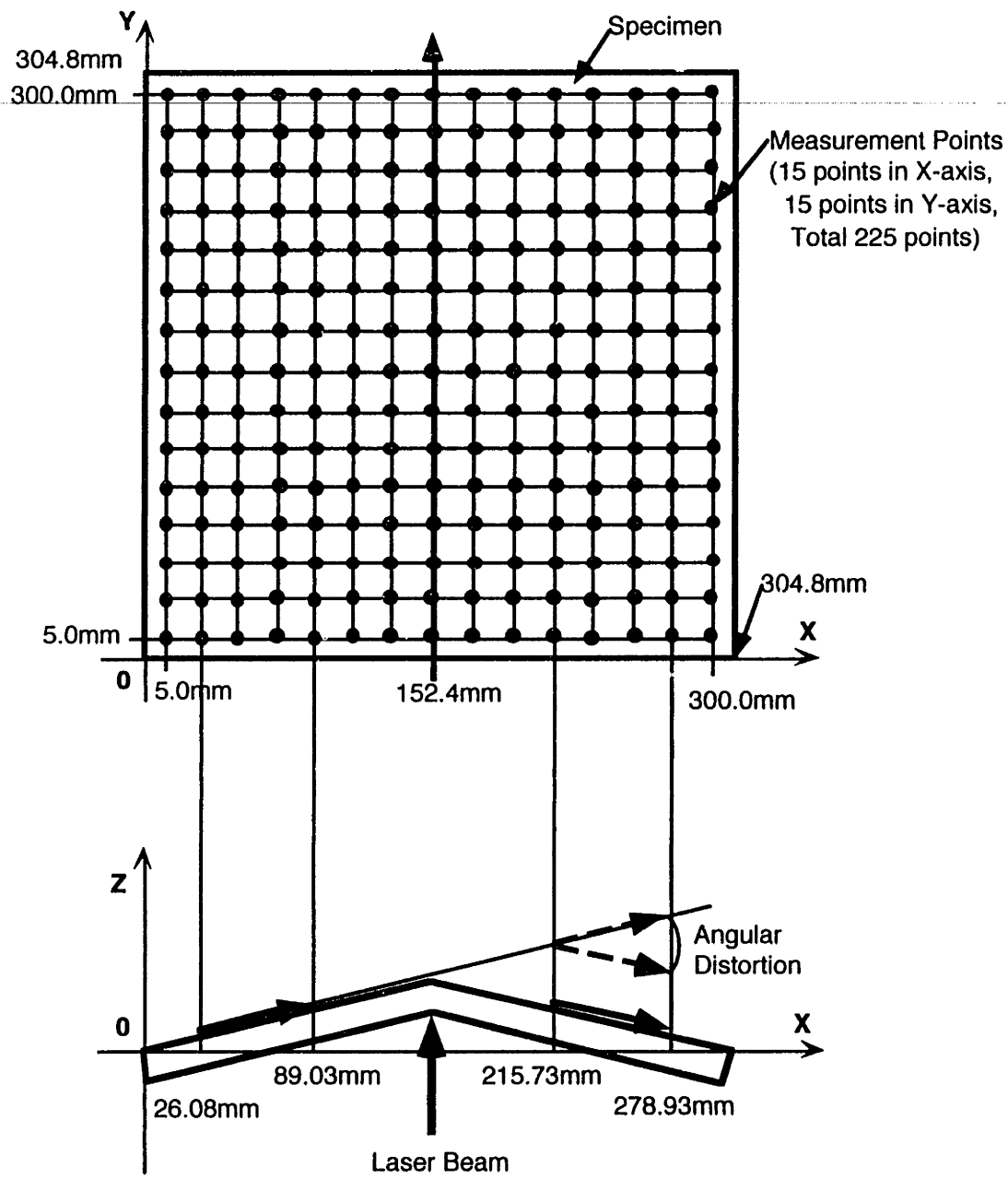


Figure B-3: Measuring setup and angular distortion

Table B.3: Process Parameters by CO_2 Laser

Specimen Ident.	Material	Thickness (mm)	Backing Condition	Coating Type	Power (W)	Spot Diameter (mm)	Velocity (mm/min)
1T	HSLA-80	6.35	aluminum	none	8,400	60	400
2T	HSLA-80	6.35	aluminum	graphite	8,400	50	400
3T	HSLA-80	6.35	none	none	8,400	50	400
4T	HSLA-80	6.35	none	graphite	8,400	60	400
5T	HSLA-80	6.35	aluminum	none	8,400	60	500
6T	HSLA-80	6.35	aluminum	graphite	8,400	50	500
7T	HSLA-80	6.35	none	none	8,400	50	500
8T1	HSLA-80	6.35	none	graphite	8,400	60	500
8T2	HSLA-80	6.35	none	graphite	8,400	60	500
9T	HSLA-80	6.35	aluminum	none	8,400	60	600
10T	HSLA-80	6.35	aluminum	graphite	8,400	50	600
11T	HSLA-80	6.35	none	none	8,400	50	600
12T1	HSLA-80	6.35	none	graphite	8,400	60	600
12T2	HSLA-80	6.35	none	graphite	8,400	60	600
INC625-1	Inconel 625	6.35	none	graphite	8,400	60	400

Table B.4: Angular Distortions by CO_2 Laser

Taguchi Run Identification	Minimum Angular Distortion (degree)	Maximum Angular Distortion (degree)	Average Angular Distortion (degree)
1T	4.6	5.66	5.02
2T	2.61	3.32	2.93
3T	4.19	5.18	4.6
4T	1.28	1.47	1.41
5T	3.94	4.69	4.37
6T	1.46	2.01	1.79
7T	3.42	4.34	3.89
8T1	1.14	1.53	1.4
8T2	0.62	0.99	0.88
9T	2.86	3.19	3.07
10T	0.58	0.95	0.8
11T	3.99	4.31	4.2
12T1	0.56	0.94	0.79
12T2	0.29	0.54	0.44
INC625-1	4.49	5.55	4.76

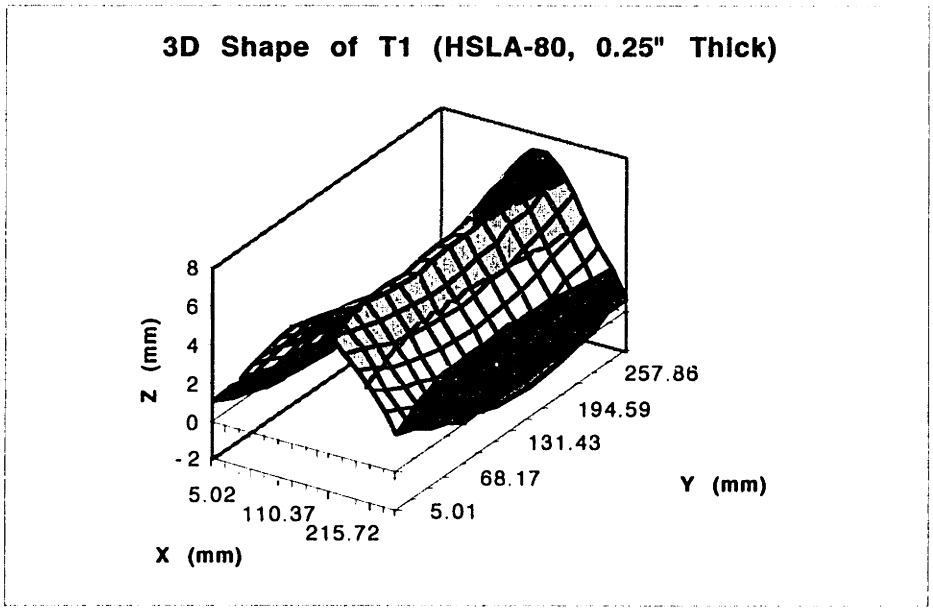


Figure B-4: Measurement result of 1T

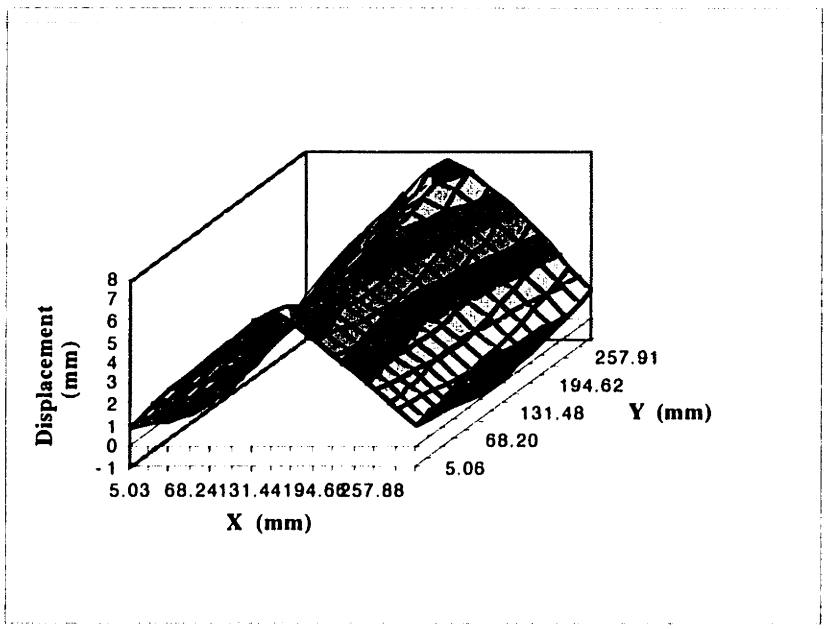


Figure B-5: Measurement result of INC625-1

Table B.5: Process Parameters by Nd:YAG Laser

Sample Ident.	Material	Thickness (mm)	Power (W)	Velocity (mm/min)	Spot Dia. (mm)	Comments
HY80-1	HY-80	7.02	2500	200	20	
HY80-2	HY-80	7.02	2500	200	20	
2219-1	A2219-O	6.35	2500	200	20	Trans. to RD
2219-2	A2219-O	6.35	2500	250	20	
2219-3	A2219-O	6.35	2500	250	20	
5052-2	A5052-O	6.38	2500	200	20	Trans. to RD
INC718-1	Inconel 718	6.55	2500	200	20	
INC718-2	Inconel 718	6.55	2500	200	20	
Ti64-1	Ti-6Al-4V	5.12	2500	200	20	Shield gas nozzle hit the specimen
Ti64-2	Ti-6Al-4V	5.12	2500	200	20	No shielding
Ti-2	Ti-15Al-3V	1.8	2500	600	20	No shielding
Ti-3	Ti-15Al-3V	1.8	1917	600	20	No shielding
Ti-4	Ti-15Al-3V	1.8	625	200	20	No shielding

Note: All specimens were coated with stove black.

B.4.2 One-Dimensional Experiment in Various Materials with Nd:YAG Laser (Experiment 4)

Six kinds of material were used in this series of experiments: HY-80, A2219-o, A5052-o, Inconel 718, Ti-6Al-4V, and Ti-15Al-3V. The spot diameter of the laser beam was kept at 20mm during the experiments. Although the laser output on the specimens by Nd:YAG laser was originally 2,600W, the output was reduced to 2,500W due to the increase of the power absorption in the fiber optics. Lower power laser beams were irradiated on thin plates of Ti-15Al-3V in order to avoid them from burning out. Faster process velocities were applied on them for the same reason. All process parameters are presented on Table B.5.

Angular distortions are shown on Table B.6. However, angular distortions could not be defined in three specimens of Ti-15Al-3V, since large amount of residual bending moment was applied along the heating lines and the profiles were not considered as folded lines as shown in Figure B-6 for example.

Table B.6: Angular Distortions in Various Materials with Nd:YAG Laser

Specimen Identification Identification	Minimum Angular Distortion (degree)	Maximum Angular Distortion (degree)	Average Angular Distortion (degree)
HY80-1	1.31	2.11	1.71
HY80-2	1.22	1.62	1.41
2219-1	1	1.13	1.07
2219-2	0.82	1.04	0.9
2219-3	1.04	1.29	1.13
5052-2	1.36	2.01	1.79
INC718-1	1.3	1.72	1.45
INC718-2	0.7	0.95	0.83
Ti64-1	1.47	2.29	1.78
Ti64-2	0.98	1.5	1.17
Ti-2	N.A.	N.A.	N.A.
Ti-3	N.A.	N.A.	N.A.
Ti-4	N.A.	N.A.	N.A.

N.A.: Angular distortion can not be measured
due to the curved surface formed by the process

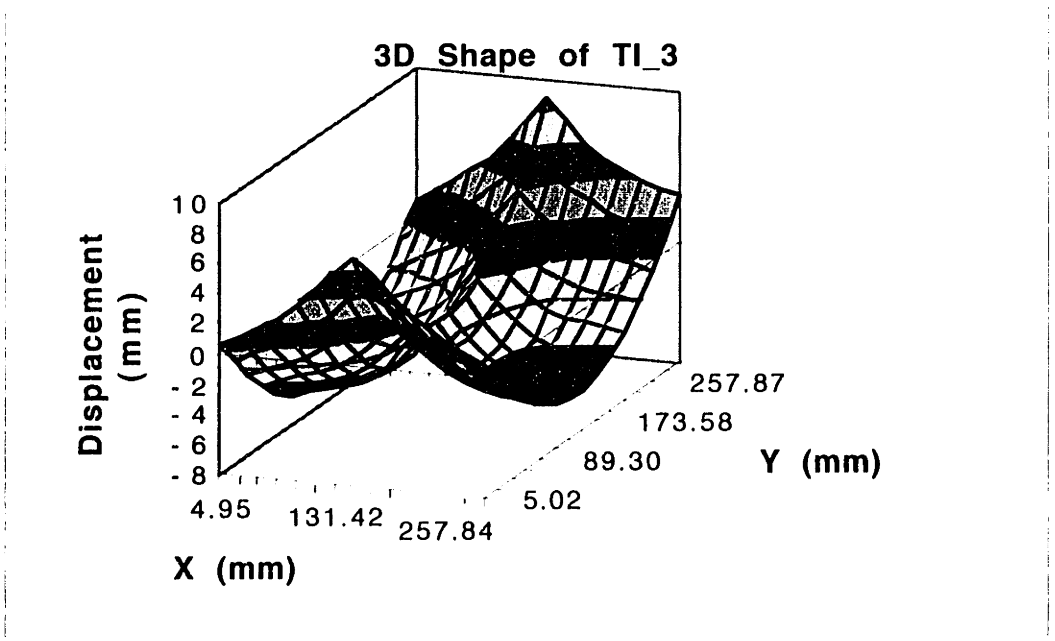


Figure B-6: Measurement result of Ti-3

Table B.7: Heat Process Conditions with Nd:YAG Laser

Sample Ident.	Material	Thickness (mm)	Orientation to Rolling Direction	Power (W)	Velocity (mm/min)	Spot Dia. (mm)
HSLAL-1	HSLA-80	6.3	Longitudinal	2,500	200	20
HSLAL-2	HSLA-80	6.3	Longitudinal	2,500	200	20
HSLAT-1	HSLA-80	6.3	Transverse	2,500	200	20
HSLAT-2	HSLA-80	6.3	Transverse	2,500	200	20
INCL-1	Inconel 625	6.3	Longitudinal	2,500	200	20
INCL-2	Inconel 625	6.3	Longitudinal	2,500	200	20
INCT-1	Inconel 625	6.3	Transverse	2,500	200	20
INCT-2	Inconel 625	6.3	Transverse	2,500	200	20
Ti64L-1	Ti-6Al-4V	5.0	Longitudinal	2,500	200	20
Ti64L-2	Ti-6Al-4V	5.0	Longitudinal	2,500	300	20
Ti64L-3	Ti-6Al-4V	5.0	Longitudinal	2,500	300	20
Ti64T-1	Ti-6Al-4V	5.0	Transverse	2,500	200	20
Ti64T-2	Ti-6Al-4V	5.0	Transverse	2,500	300	20
Ti64T-3	Ti-6Al-4V	5.0	Transverse	2,500	300	20

B.4.3 Investigation of the Effect of Rolling Directions (Experiment 5)

The objective of the experiments was to investigate the effects of rolling direction (RD) on the angular distortions. The experiments were conducted in three kinds of material: HSLA-80, Inconel 625, and Ti-6Al-4V. The same set of heat process parameters were used for two kinds of orientation: parallel to the RD (Longitudinal) and perpendicular to the RD (Transverse). Laser power was kept at 2,500W and its spot diameter was kept at 20mm during the experiments. Process velocities were mainly 200mm/sec and faster velocities of 300mm/sec were applied for Ti-6Al-4V plates in order to increase the angular distortion.

In order to investigate the effects of RD closely, each process condition including the orientation of the plate was repeated twice.

Angular distortions obtained in the series of the experiments are shown on Table B.8. We can observe the slight effect of RD on the angular distortion in HSLA-80. Angular distortions caused with the heating lines parallel to RD were slightly larger than those with the heating line perpendicular to RD. However, since the discrepancy of angular distortion among the same process conditions are relatively large, it

Table B.8: Angular Distortions with Nd:YAG Laser

Specimen Identification	Minimum Angular Distortion (degree)	Maximum Angular Distortion (degree)	Average Angular Distortion (degree)
HSLAL-1	1.09	1.39	1.25
HSLAL-2	1.39	1.55	1.46
HSLAT-1	1.89	2.51	2.2
HSLAT-2	1.45	1.73	1.6
INCL-1	3.78	4.38	4.03
INCL-2	2.55	3.84	2.96
INCT-1	3.21	4.04	3.48
INCT-2	3.9	4.17	4.01
Ti64L-1	0.96	1.36	1.17
Ti64L-2	2.37	3.42	2.74
Ti64L-3	3.06	3.72	3.23
Ti64T-1	1.32	1.91	1.53
Ti64T-2	2.5	3.33	2.76
Ti64T-3	3.04	3.72	3.25

is difficult for us to confirm the effect of RD on the angular distortion.

On the other hand, apparently the effect could not be observed in Inconel 625 and Ti-6Al-4V in these experiments.

B.5 Discussion

The final shapes of specimens, angular distortions in various materials, and the effect of rolling directions on angular distortion are discussed in this section.

B.5.1 Final Shape of Specimen

Transverse and longitudinal profiles to the heating line and the distribution of angular distortion were examined in terms of the final shapes of specimens.

Transverse Profiles to the Heating Line

All the data plot within the x-axis range of 26mm to 89mm and 216mm to 279mm are on the line for all plates except for thin titanium plates: Ti-2, Ti-3, and Ti-4.

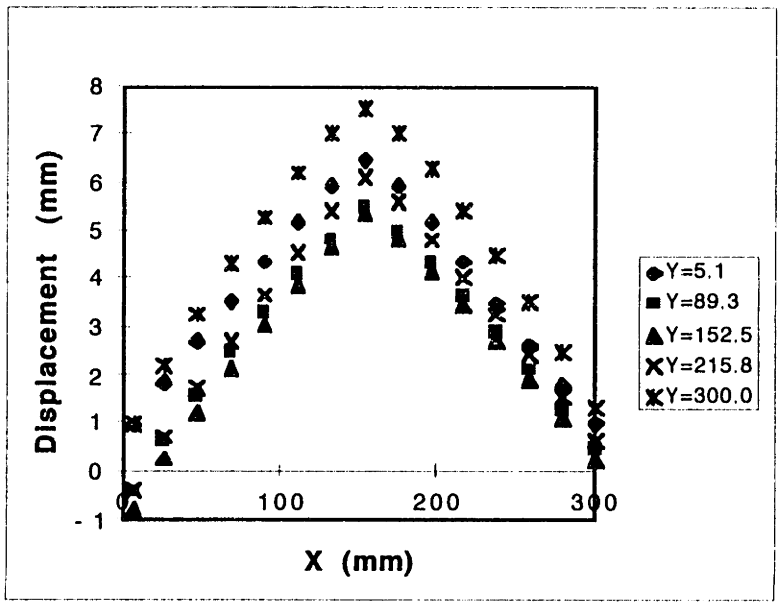


Figure B-7: Transverse profiles of INC625-1

These results validate the use of angular distortions as the representations of final shapes. As a example, the transverse profiles of INC625-1 are shown in Figure B-7.

Distribution of Angular Distortion

Angular distortions caused by the heat process change over the heating line as shown in Figure B-8. Almost all plate processed in the experiment have tendency that the angular distortion increase from the start point to the end point of the process.

Longitudinal Profiles to the Heating Line

The plate processed by a laser beam were slightly bent in the direction perpendicular to the heating line as shown in Figure B-9. Since this bent over was observed in the numerical result by a linear FEM program with the pure bending moment perpendicular to a heating line applied in the middle of a plate, the bent-over longitudinal profiles seem to be caused by the bending moment perpendicular to the heating line by a heat process.

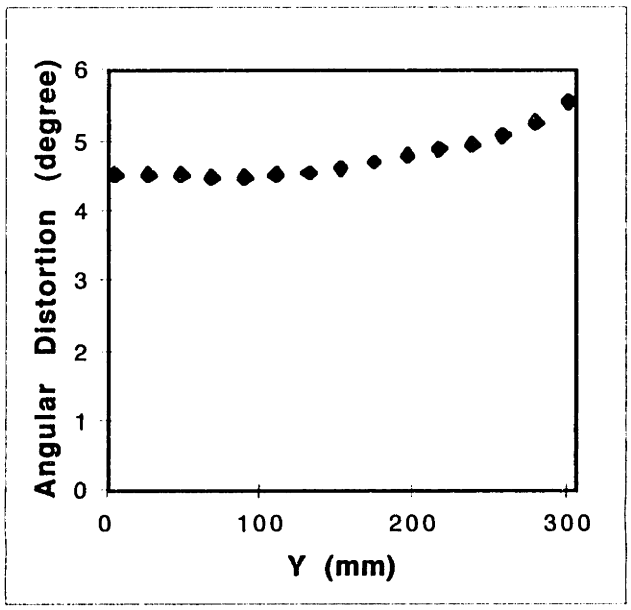


Figure B-8: Distribution of angular distortion in INC625-1 along the y-asxis

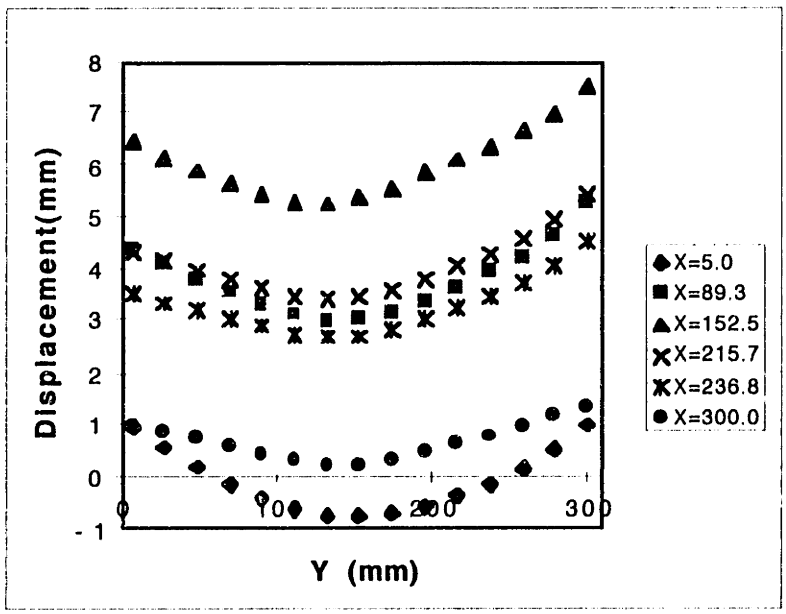


Figure B-9: Longitudinal profiles of INC625-1

B.5.2 Angular Distortion in Various Materials

All average angular distortions in all the experiments are plotted in Figure B-10 in order to examine the relationship between angular distortion and the parameter which was used for results of CO_2 laser forming by Masubuchi. The parameter is expressed as follows:

$$\frac{P}{t \cdot \sqrt{v}} \qquad [\frac{kJ}{sec^{0.5} \cdot mm^{1.5}}] \tag{B.1}$$

Where P is the power of a laser beam in W, t is thickness of a plate in mm, v is the velocity of the heating process in mm/sec.

From the figure, we cannot clearly see any relationship between angular distortion and the parameter due to the deviation of data. We can observe a slight tendency that the angular distortion increases with the increase of the parameter.

However, the angular distortion in Ti-6Al-4V decreases with the increase of the parameter. This is due to the deep heat penetration of Ti-6Al-4V plates. When the process velocity is large or the parameter is small, temperature gradient is produced along the thickness. On the other hand, when the process velocity is small or the parameter is large, no temperature gradient is produced since the heat conduction rate is small.

B.5.3 Effect of Rolling Directions on Angular Distortion

The effect of rolling directions on angular distortion is investigated in three kinds of material: HSLA-80, Inconel 625, and Ti-6Al-4V. The data of angular distortions are plotted in percentile charts in Figure B-11, B-12, and B-13. A percentile chart represents each variables plotted as a separate box. The y-axis displays the range of the data of angular distortions and the x-axis displays the specimens. Each box encloses 90% of the data. The bottom and top of each box represent 5% and 95% of the data. Three lines are drawn inside each box. The middle line represents the median value of the data (50%), while the lower and upper dashed lines represent 25% and 75% of the data, respectively.

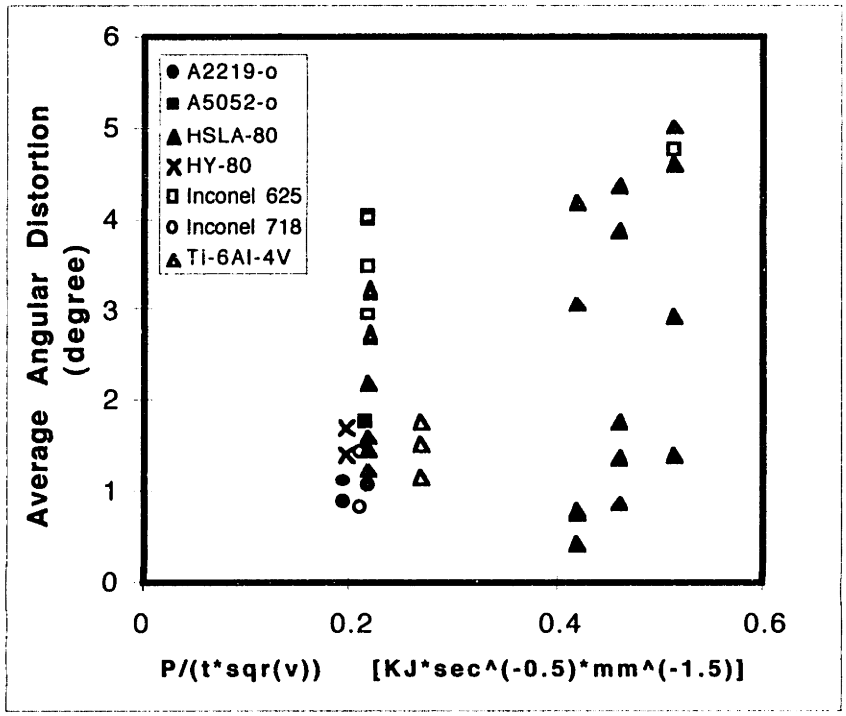


Figure B-10: Results of parametric study

Four specimens processed with the same heating conditions are shown in each figure. The left two specimens were processed with the rolling direction parallel to the heating line (longitudinal position) and the right two specimens were processed with the rolling direction perpendicular to the heating line (transverse position).

It is difficult to observe the effect of rolling directions on angular distortion due to the deviation of the data itself. Apparently, there is no effect of rolling directions in Inconel 625 and Ti-6Al-4V. There seems to be effect of rolling directions in HSLA-80. The angular distortions produced with the transverse position process are slightly larger than those produced with the longitudinal position process.

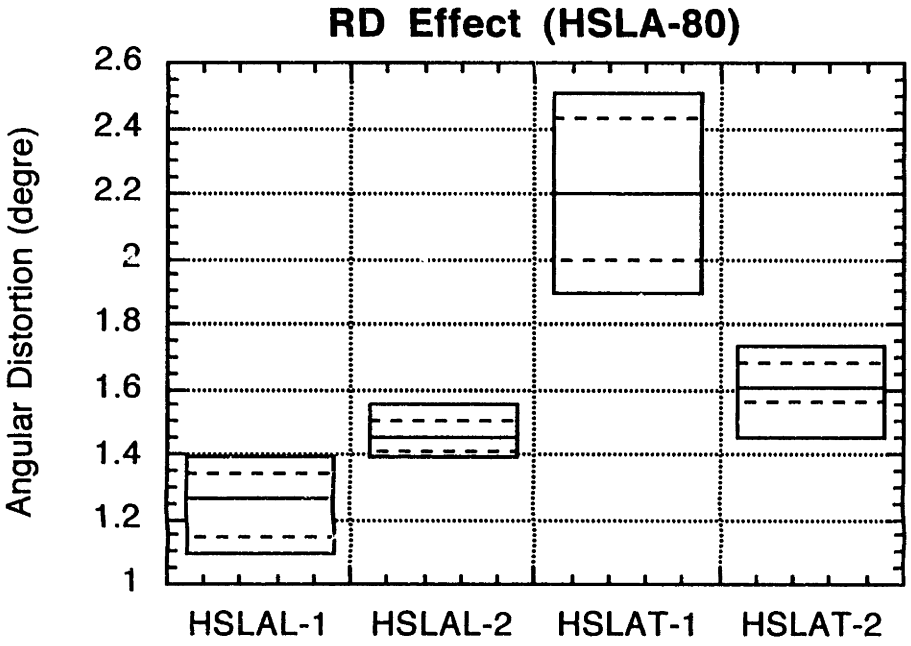


Figure B-11: Effect of rolling directions on angular distortion in HSLA-80

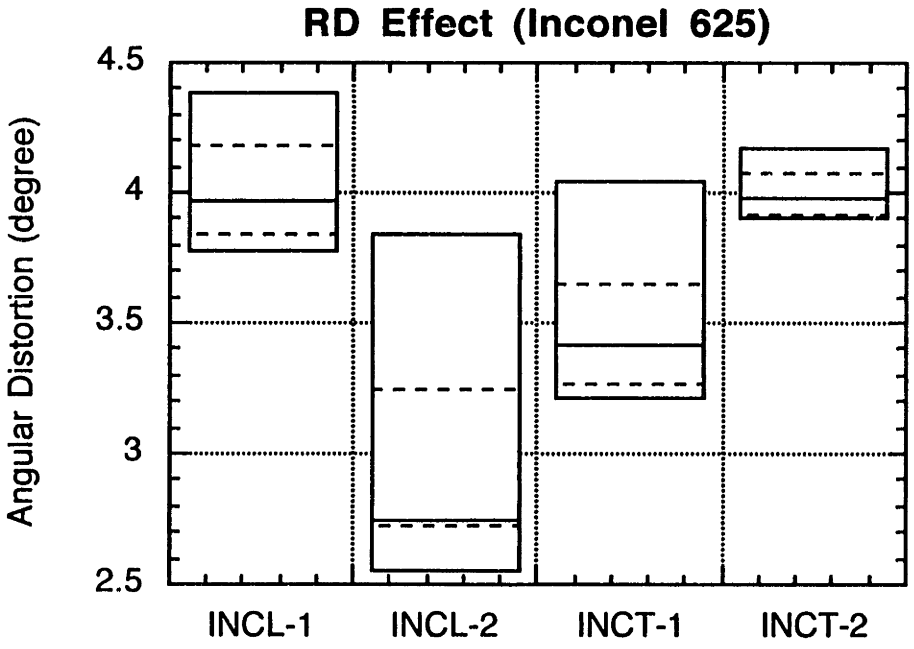


Figure B-12: Effect of rolling directions on angular distortion in Inconel 625

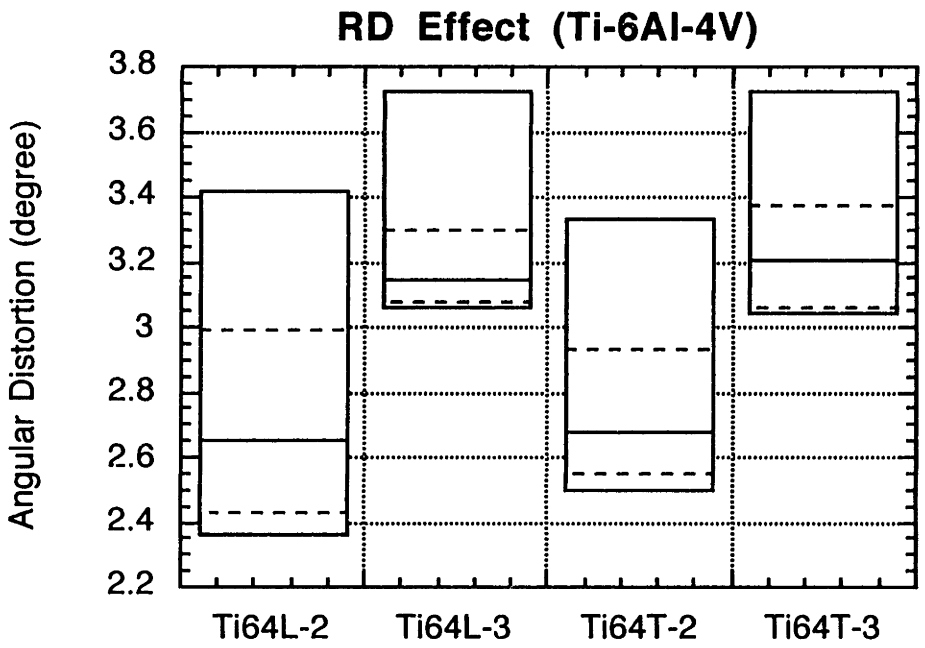


Figure B-13: Effect of rolling directions in Ti-6Al-4V

B.6 Summary

The results of final shape measurement were presented. The final shapes were measured by a CMM. The displacements from the base plane of the CMM were measured at 225 points in each plate. Angular distortions were defined with four displacements in an intersection perpendicular to a heating line. Fifteen angular distortions were calculated on each specimen.

The use of angular distortion as the representation of a final shape was validated by close examination of transverse profiles to a heating line for most specimens. The transverse profiles are considered as folded lines. However, when angular distortion is relatively large compared to the thickness of a plate, the transverse profiles are no longer folded lines as seen in the thin plate of Ti-15Al-3V.

Longitudinal profiles of processed plates were slightly convex toward the heating lines. These convex shapes seemed to be caused by the bending moment caused by heating lines and distributing along the heating lines.

The relationship between the angular distortion and the process parameter proposed by Prof. Masubuchi at MIT has been investigated using results obtained in the series of the experiment. Since the range of the process parameter was relatively small and the deviation of angular distortions were large, the clear relationship could not been observed.

The effect of rolling directions on the angular distortion was not observed in Inconel 625 and Ti-6Al-4V. In HSLA-80, angular distortions produced with rolling directions in transverse position to a heating line were slightly larger that those produced with rolling direction in longitudinal position. These results should be re-examined with more results later.

Appendix C

Transient Angular Distortion Measurement of Laser Forming

C.1 Introduction

Laser line heating experiments were conducted at the Applied Research Laboratory, Pennsylvania State University (ARL Penn State) from June 11 to June 13, 1996, to obtain thermal and displacement data for comparison to model predictions. Although the thermal data have been presented already by Mr. Martukanitz and Mr. Sumnicht at ARL Penn State on July 12, 1996, we need to calculate transient angular distortions in order to compare the experimental results to the numerical results. This report presents the results of transient angular distortions calculated from the transient displacement data during and after the process. These data enable us to validate the computational model directly.

Also, some investigations that try to clarify the effect of the multi-pass process on deformation, the effect of initial shape of a plate and the residual stresses are presented in the report. The results of the investigations show the effectiveness of using a linear FEM analysis in the optimum procedure of heating conditions.

C.2 Experimental Setup

C.2.1 Specimen

We used two materials; Inconel 625 and HSLA-80, in the same size of 304.8 mm (12") × 304.8 mm (12") × 6.35 mm (0.25").

C.2.2 Laser Process Apparatus

The experiments were conducted in the Applied Research Laboratory at the Pennsylvania State University. A 3 kW Nd:YAG laser manufactured by HOBART TAFA was used. A laser beam generated by the oscillator was transferred by an optical fiber tube to a heat process head at the site. Although the energy of the laser beam was dispersed in the fiber tube, the output of the oscillator was adjusted to keep the output on a specimen 2.6 kW constantly.

C.2.3 Displacement Sensors

Laser Displacement Sensor

Non-contact laser displacement sensors manufactured by KEYENCE were used to measure the vertical displacement of the specimen during heat processes. The combination of a sensor head, LB-081, and a controller, LB-1101, has a measuring range of ±7.5 mm, linearity of 0.25 % of full scale, resolution of 8 μm, and response frequency of 36 Hz.

Displacement Transducer

Contact displacement transducers, model number LD300-150, manufactured by OMEGA were also used in the experiment. The excitation voltage was 10 VDC.

C.2.4 Data Acquisition System

We used DAKBOOK 100 fabricated by OMEGA as a data acquisition system.

C.2.5 Measuring Setup

We used two measuring setups, setup A and setup B, in the experiments. Figure C-1 shows those setups. First, we used setup A. Then we realized that measurements at positions 7 and 8 given by displacement transducers contained much noise. Also, we realized that measuring data at positions 1 and 2 seldom moved vertically. Then, we replaced displacement transducers by laser displacement sensors at positions 7 and 8. Although displacement transducers still gave us much noise during measurement, ignoring the displacement at positions 1 and 2, setup B gave much better data than setup A did.

We set a coordinate system so that X-Y plane is parallel to the top and bottom surfaces of a specimen. Z-axis was defined in the vertical direction so that positive direction pointed upward.

The specimens were clamped to a support at the origin of the coordinate system so that we could put sensors beneath the specimens.

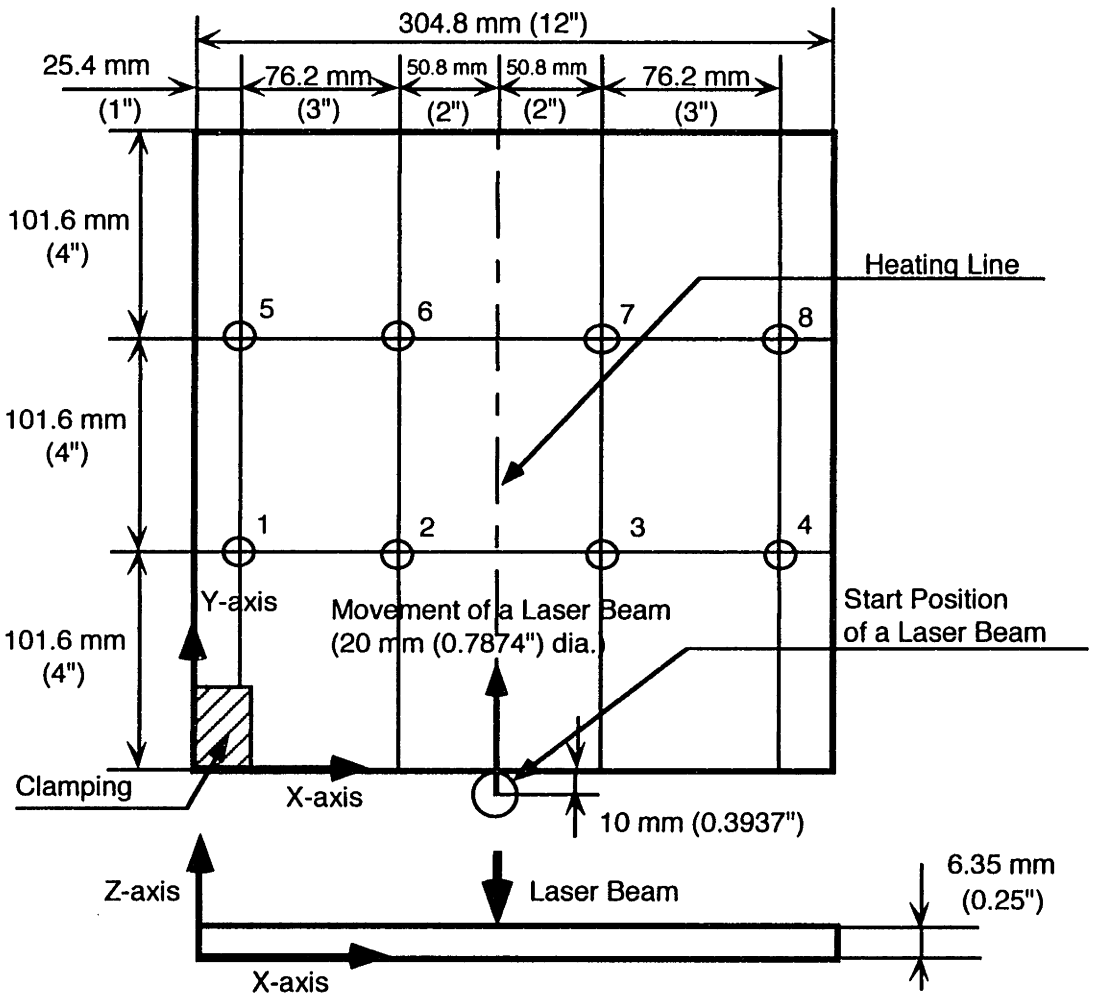
C.3 Heat Process Conditions

C.3.1 Inclination Angle and Stand-off Distance

We used a Nd:YAG laser beam as a moving heat source. A laser head has an inclination angle to protect the laser system from the reflection of the laser beam itself. Figure C-2 shows the definition of the inclination angle in a heat process.

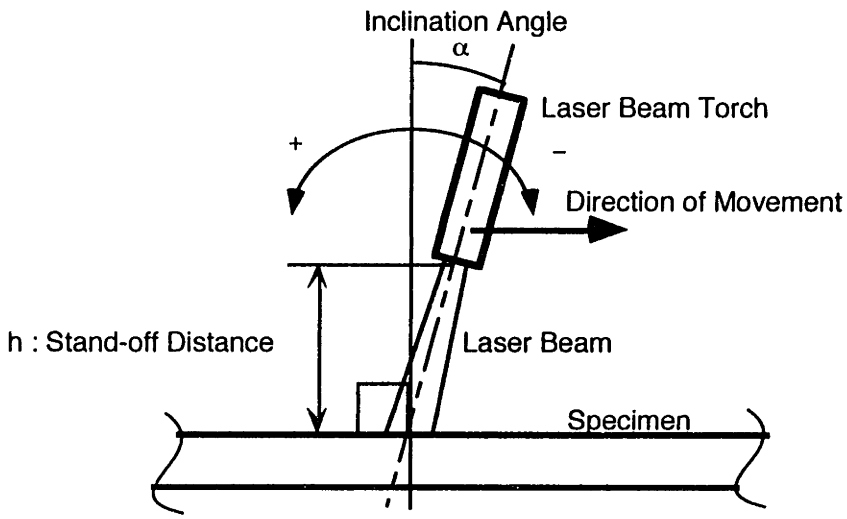
C.3.2 Constant Heat Process Conditions

We used a constant heat process conditions as shown in Table C.1. All displacement measurements started one second before the start of the laser beam movement. Just after the start of the movement the laser beam start contacting with a plate. The laser beam kept contacting with a plate while it traversed the length of the plate and the diameter of the laser beam. Consequently, the duration time of one process was $(304.8 + 20)/200 \times 60 = 97.44$ sec each. In order to obtain the final displacements

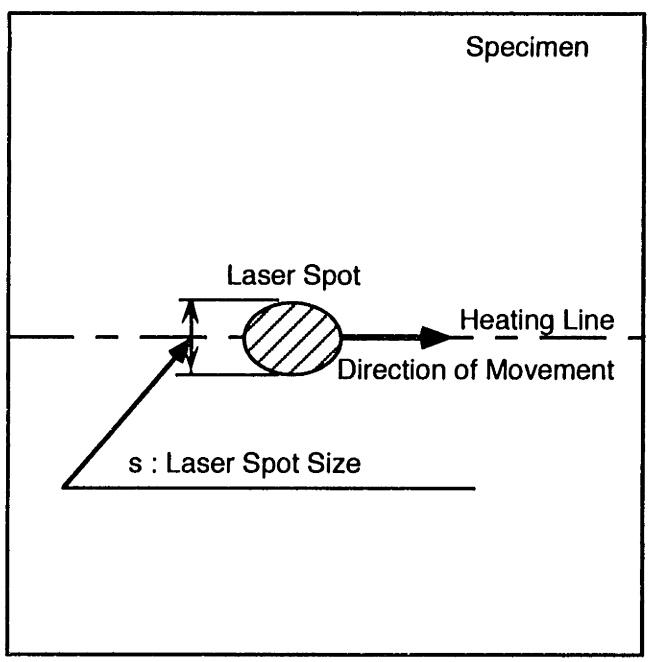


Position	Setup A	Setup B
1	Laser Sensor (L1)	Displacement Transducer (TR1)
2	Laser Sensor (L2)	Displacement Transducer (TR2)
3	Laser Sensor (L3)	Laser Sensor (L5)
4	Laser Sensor (L4)	Laser Sensor (L6)
5	Laser Sensor (L5)	Laser Sensor (L1)
6	Laser Sensor (L6)	Laser Sensor (L2)
7	Displacement Transducer (TR1)	Laser Sensor (L3)
8	Displacement Transducer (TR2)	Laser Sensor (L4)

Figure C-1: Measuring setup



(a) Inclination Angle (α) and Stand-off Distance (h) in the Side View of a Process



(b) Laser Spot Size (s) in the Top View of a Process

Figure C-2: Definitions of parameters of a heat process

Table C.1: Heat Process Conditions Process

Process Parameter	Value
Output at a Laser Oscillator	3.0 kW
Output of a Laser Beam on a Specimen (P)	2.6 kW
Stand-off Distance (h)	190 mm
Laser Spot Size (s)	20 mm (.787 inch)
Inclination Angle (a)	-10 degree
Traverse Rate (u)	200 mm/min (7.87 ipm)

Table C.2: List of All Heat Processes Conducted in the Experiment

Specimen ID	Material	Process ID	Setup	Process Description
IN625-1	Inconel 625	IN625-1	A	One pass in the middle
IN625-2	Inconel 625	IN625-2	A	One pass in the middle
IN625-3	Inconel 625	IN625-3	A	One pass in the middle
IN625-4	Inconel 625	IN625-4	A	One pass in the middle
		IN625-4B	B	Second pass perpendicular to the heating line of IN625-4
IN625-5	Inconel 625	IN625-5	A	One pass in the middle
IN625-6	Inconel 625	IN625-6	A	One pass in the middle
		IN625-7	A	Second pass on the heating line of IN625-7
IN625-8	Inconel 625	IN625-8	B	One pass in the middle
H3D	HSLA-80	H3D-1	B	One pass in the middle
		H3D-2	B	Second pass perpendicular to the heating line of H3D-1

of plates, the duration times of the measuring were set either 300 sec or 600 sec. All plates were cooled naturally during the measurement of the displacement data.

C.3.3 Heat Processes

We have processed the following eight plates shown in Table C.2. Since the second heating lines were put onto some plates, we have conducted eleven heat processes.

C.4 Results

C.4.1 Definition of Angular Distortion

Although we measured vertical displacements at eight points on each plate during a heat process, dealing with these eight data individually is not convenient for us to

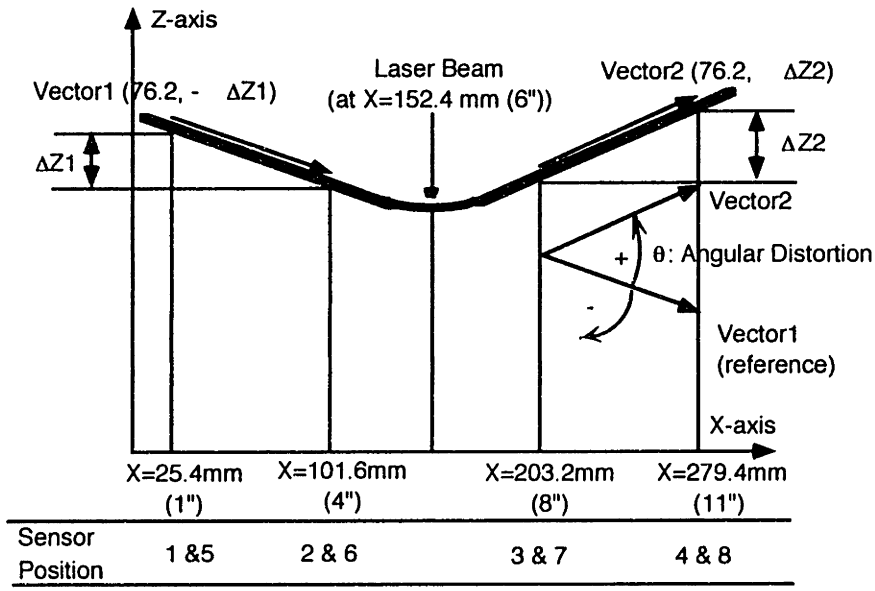


Figure C-3: Definition of angular distortion

understand the phenomena during the heat process. The measuring setup is designed to calculate two angular distortions perpendicular to the heating line. The data at position 1 to 4 gives the angular distortion at y=101.6 mm and the data at position 5 to 8 gives that at y=203.2 mm. The definition of angle distortion is shown in Figure C-3.

In Figure C-3, vector 1 is a reference vector for angular distortion. As the angular distortion θ is defined positive in the counter clockwise direction, positive values of θ show convex shapes of a plate and negative values show concave shapes.

Practically, the angular distortion, θ, is given by the following equation.

$$\theta = \frac{\Delta Z_2 - \Delta Z_1}{|\Delta Z_2 - \Delta Z_1|} \arccos\left(\frac{\vec{v_1} \cdot \vec{v_2}}{|\vec{v_1}||\vec{v_2}|}\right) \tag{C.1}$$

In Equation C.1, the first term decides the sign of the angular distortion.

C.4.2 Results of Transient Angular Distortion

Comparison between Two Setups

Figure C-4 shows the typical characteristics of measuring data by setup A. Two transducer displacement sensors and two laser sensors were used to measure the displace-

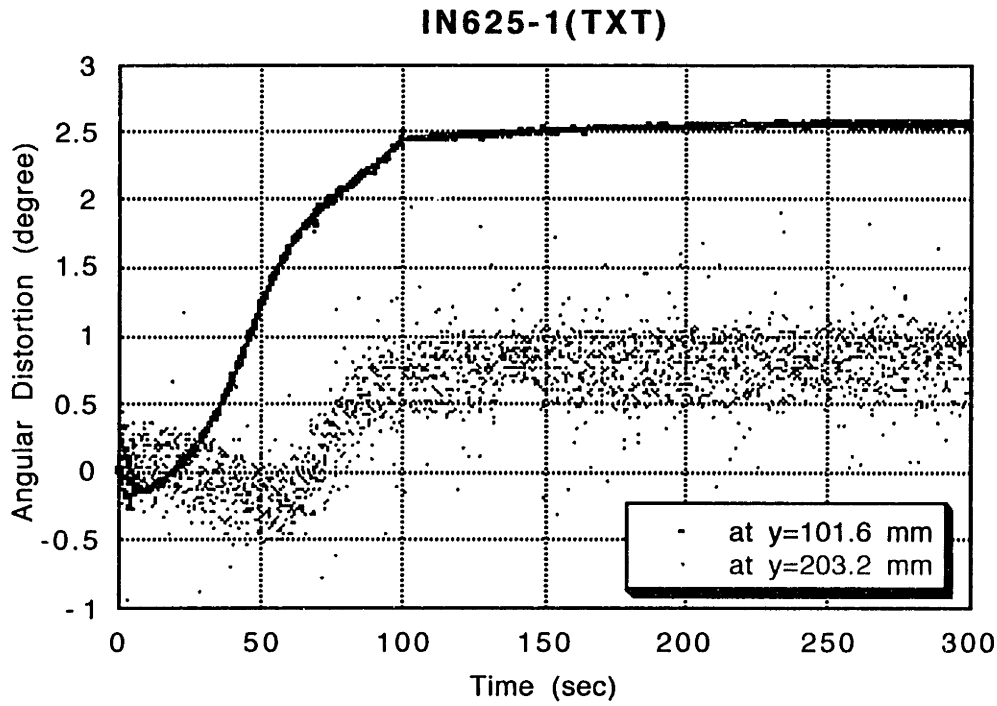


Figure C-4: Example of results measured by setup A

ment at y=203.2 mm in the setup A. As the signals from transducer displacement sensors contain much noise, we obtained meaningless data for the transient angular distortions at y=203.2 mm by the setup A.

Then, we switched from setup A to setup B. Figure C-5 shows its characteristic. Although two transducer displacement sensors were used to measure the displacement at y=101.6 mm and the range of the fluctuation of their signals was the same as that of the example shown in Figure C-4, the transient angular distortion data are much more meaningful. This is because the angular distortion is not sensitive to the displacement data measured by transducer sensors.

C.4.3 Neglecting Transducer Sensor Data

The displacement data at position 1 and 2 were nearly zero at all times, since these positions were close to the support point. The actual values of the displacement measured by the sensors were always the range of 10 - 2 mm. Then, only two data

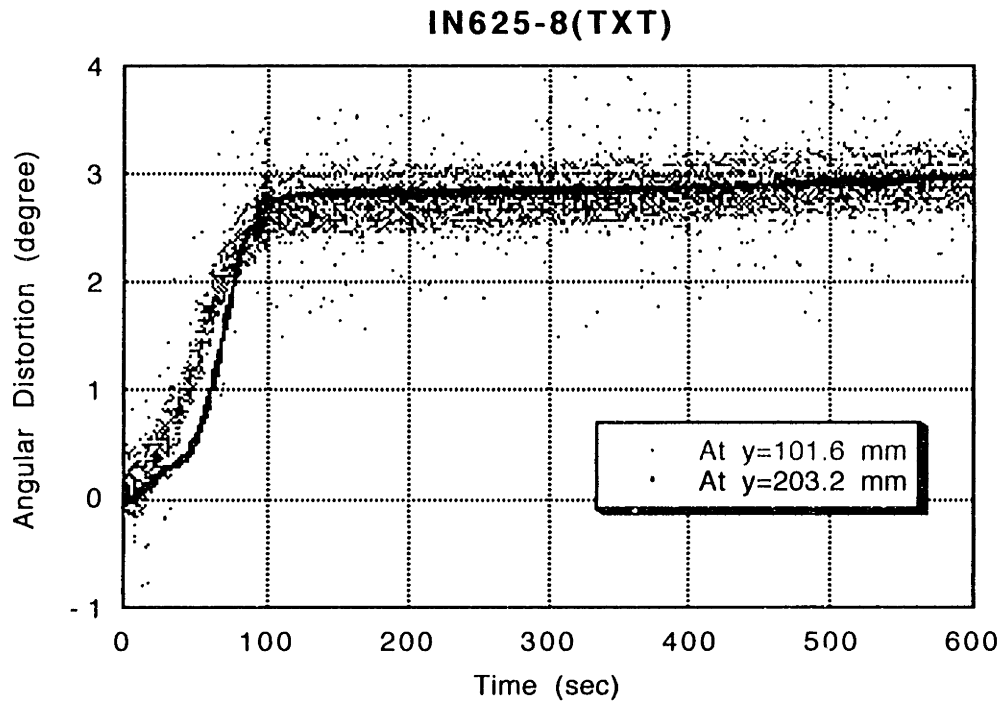


Figure C-5: Example of results measured by setup B

measured at position 3 and 4 by laser sensors were used to obtain the angular distortion at y=101.6 mm in Figure C-6.

The transient angular distortion at y=101.6 mm went negative at the beginning, started increasing after approximately 10 seconds of the measuring, and almost reached to the final distortion at the end of the heat process.

The transient angular distortion at y=203.2 mm gradually increased from the beginning up to 50 seconds after the start of the heat process. When the laser beam approached to the point at y=203.2 mm (it passed there at 60 seconds), the distortion started increasing greatly and finished increasing at the end of the process.

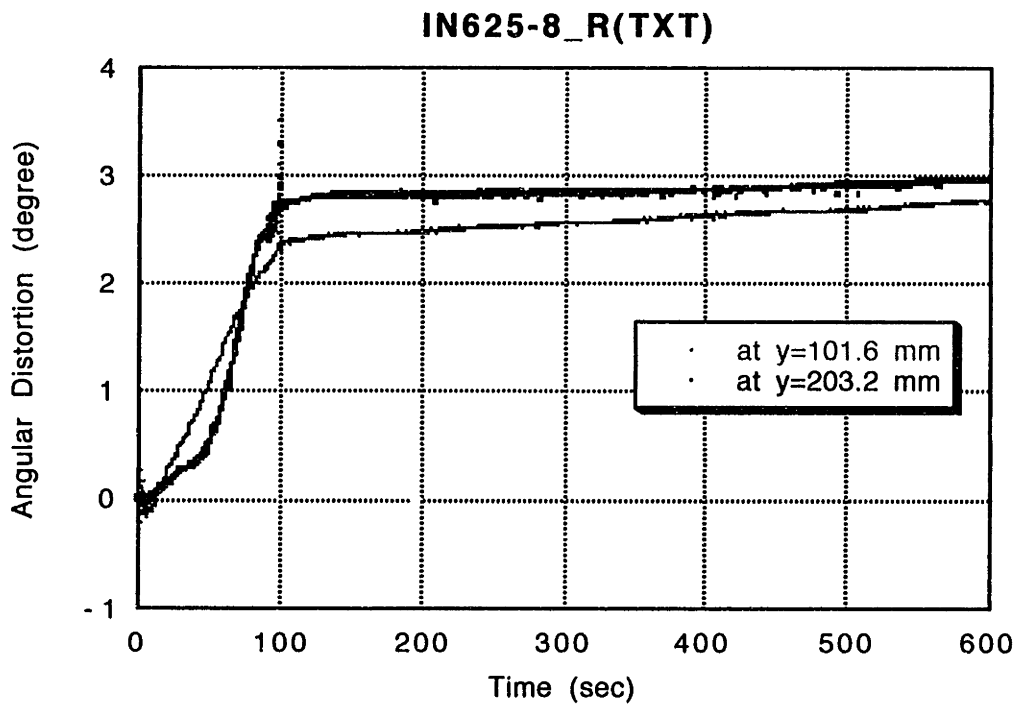


Figure C-6: Example of results with neglecting data measured by transducer displacement sensors

C.5 Discussion

C.5.1 Repeatability of Deformation

In the experiments we obtained seven transient angular distortion data at y=101.6 mm. These six data are super-imposed in Figure C-7. Qualitatively the transient angular distortions are the same and the figure shows their good repeatability.

All data have the same tendency as explained at Section C.4.3. They decreased in 10 seconds after the starts of the processes, then started increasing, and continued increasing by the end of the process. All the increases of the angular distortions finished at the ends of the processes.

The actual values, the average value, and the standard deviation of the final angular distortion are as follows:

- Actual values: 2.56, 2.85, 2.83, 2.90, 2.64, 2.51, 2.63
- Mean value: 2.70
- Standard deviation: 0.1559

C.5.2 Comparison with the Final Shape

The last values of angular distortions measured by laser displacement sensors are compared with the values at the final shape in Table C.3. The angular distortions were measured by a coordinate-measuring machine at MIT using the exact same position as the transient measurement.

The angular distortions in the final shapes of the specimens have been reported from the ARL Penn State on July 12, 1996. However, these values were measured by an simple and less-accurate method. The values shown in Table C.3 are very accurate.

In Table C.3, the last values of transient measurement are slightly smaller then the values measured in the final shape.

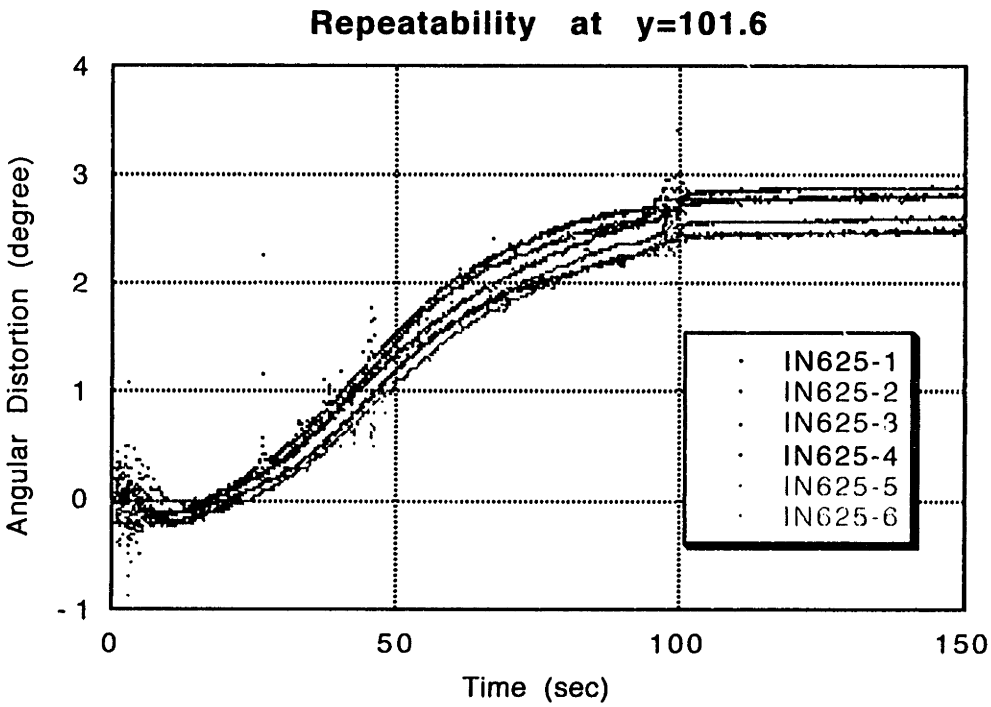


Figure C-7: Comparison among six transient angular distortions

Table C.3: Comparison in Angular Distortions between the Last Values of Transient Measurement and the Final Shape

Specimen ID	Position	Last Value of Transient Measurement (degree)	Value at the Final Shape (degree)
IN625-1	Y=101.6mm	2.56	3.80
	Y=203.2mm	N.A.	4.10
IN625-2	Y=101.6mm	2.88	3.60
	Y=203.2mm	N.A.	3.90
IN625-3	Y=101.6mm	2.87	3.50
	Y=203.2mm	N.A.	4.00
IN625-5	Y=101.6mm	2.7	2.75
	Y=203.2mm	N.A.	2.85
IN625-8	Y=101.6mm	2.76	3.40
	Y=203.2mm	2.96	3.70

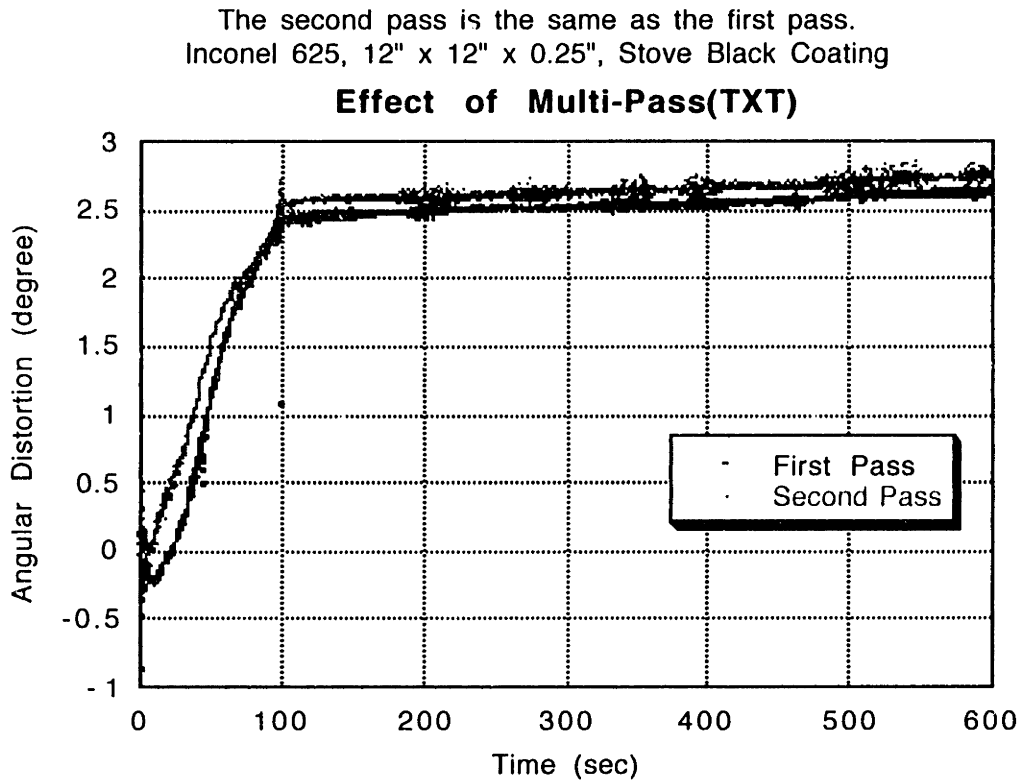


Figure C-8: Comparison between transient angular distortions during the first pass and the second pass at the same line

C.5.3 Effect of Multi-Pass

Two heating lines were put on the same position onto specimen IN625-6. The angular distortions by the first pass and the second pass both at y=101.6 mm are shown in Figure C-8. Although the movements at the beginning of the processes were slightly different each other, the final values were the same. Figure C-8 shows that superimposability holds for multi-pass deformation at least for two passes.

C.5.4 Effects of the Initial Shape and Residual Stresses

Figure C-9 shows the comparison of the angular distortions at y=101.6 mm in an Inconel 625 plate that were caused by the first pass in the middle of a plate and the second pass perpendicular to the first pass. The transient angular distortion by the second pass was almost the same as that by the first pass. However, it decreased

largely just before the end of the process and kept decreasing while the plate was cooled. Consequently, the final value of angular distortion by the second pass was smaller than that by the first pass by 14%.

Similarly, the comparison of the angular distortions in an HSLA-80 plate are shown in Figure C-10. In the figure, both angular distortions decrease just before the ends of the processes. All Inconel 625 plates were cut by a mechanical shear to form squares. On the contrary, all HSLA-80 plates were cut by some thermal cutting method, either gas frame cutting or plasma cutting. Therefore, the HSLA-80 plates had significantly larger residual stresses. The decrease of the angular distortions in the HSLA-80 plate might be caused by the residual stresses induced by thermal cutting.

The above mentioned cause of the decreases in the HSLA-80 plate is supported by the decrease of the angular distortion by the second pass shown in the Inconel 625 plate. In the Inconel 625 plate, when the first pass was started, the plate was almost residual stress free. After the first pass, the residual stresses were induced in the plate and these residual stresses caused the decrease of the angular distortion just before the end of the second process.

We can observe the slight declinations of the angular distortions by the second pass in the HSLA-80 plate after the heat process. There is a possibility that the declination was caused by the initial shape formed by the first pass. The same declination of the deformation by the second pass is observed in the Inconel 625 plate as shown in Figure C-9.

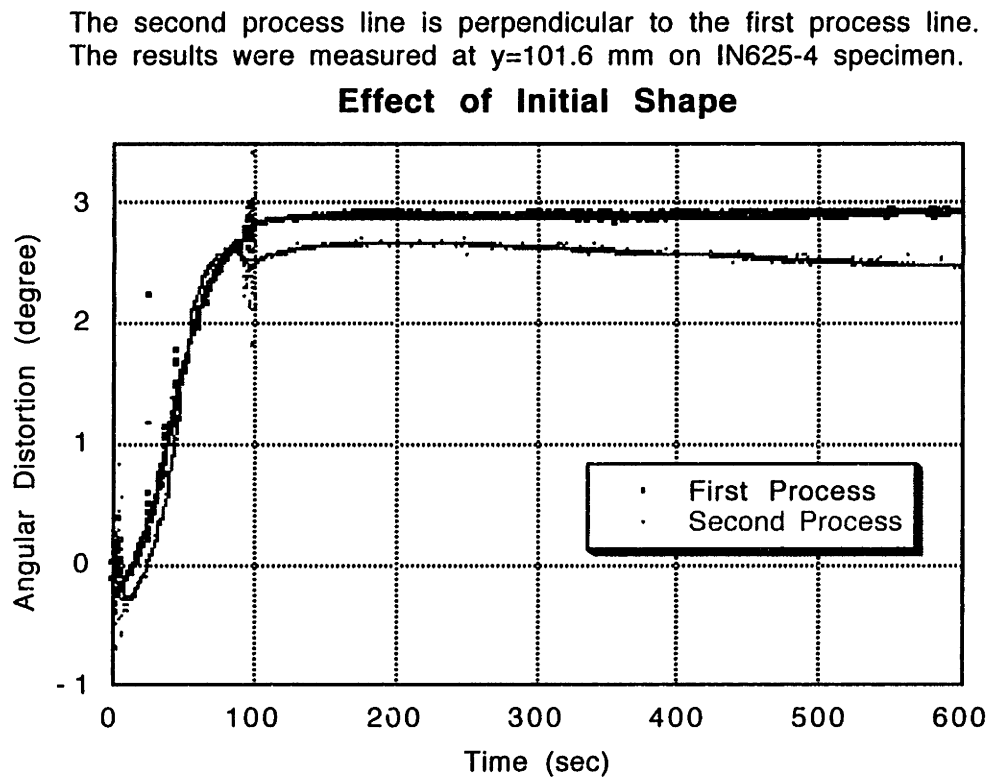


Figure C-9: Effect of initial shape before the process in an Inconel 625 plate (the second pass is perpendicular to the first pass)

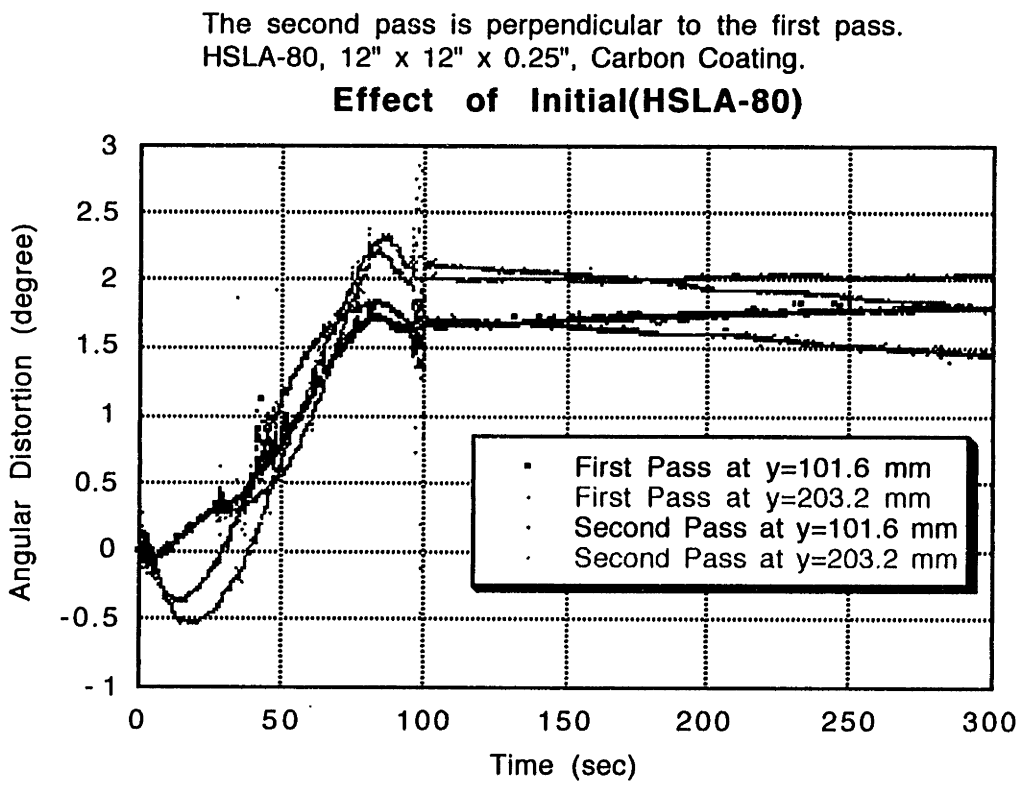


Figure C-10: Effect of initial shape and residual stress in an HSLA-80 plate (the second pass is perpendicular to the first pass)

C.6 Summary

Measurement data of the transient displacements were transformed to the transient angular distortion in order to enable the comparison between the experimental results and the numerical data. Using the transient angular distortion, we can validate the numerical models under development.

Also, the investigations were conducted to understand the mechanism of the deformation of a plate by laser line heating. One of the investigations shows that the shape of a plate near the starting point of a process becomes convex before the beam from the top passes the closest point to the measurement line. Then the shape becomes concave. On the contrary, the shape of the plate near the end point of the process becomes concave gradually rather than convex when the beam approaches. Then, after the beam passes by, the rate of the deformation increases.

The effects of the initial shape and the residual stress on the deformation by laser line heating were investigated. The deformation in an Inconel 625 was decreased by approximately 14% with the initial shape and the residual stress induced by the first pass of laser line heating in the experiment. Also, the deformation in an HSLA-80 was decreased by about 17%. These slight effects of the initial shape and the residual stress allow us to utilize a linear FEM model to predict the complicated shape by many heating lines.

Bibliography

[1] Muturo Araki, Nobuo Inoue, Masaki Horioka, and Masaaki Ando. On angular distortion of hull steel plates by line heating methods. *Journal of the Society of Naval Architects of Japan*, 133:343–348, June 1973. (in Japanese).

[2] K.J. Bathe. ADINAT - a finite element program for automatic dynamic incremental nonlinear analysis of temperature. Technical Report 82448-5, Department of Mechanical Engineering, Massachusetts Institute of Technology, May 1977.

[3] L. K. Biegeleisen. Residual stresses and deflections in complex shapes formed by laser bending. S.M. thesis, Department of Ocean Engineering, Massachusetts Institute of Technology, Cambridge, Mass., 1986.

[4] Bruno A. Boley and Jerome H. Weiner. *Theory of Thermal Stresses.* John Wiley & Sons, Inc., New York, U.S.A., 1960.

[5] David L. Carroll. Fortran genetic algorithm (GA) driver. http://www.staff.uiuc.edu/ carroll/ga.html, January 1997. Home Page on the Internet.

[6] D. L. Deacon. Material degradation in heavy steel plates caused by bending with a laser. S.M. thesis, Department of Ocean Engineering, Massachusetts Institute of Technology, Cambridge, Mass., 1984.

[7] M. Geiger. Synergy of laser material processing and metal forming. *Annals of the CIRP*, 43(2):563–570, 1994.

[8] M. Geiger and F. Vollertsen. Mechanism of laser forming. *Annals of the CIRP*, 42(1):301–304, 1993. (in German).

[9] Manfred Geiger, Jürgen Kraus, and Frank Vollertsen. Laser beam forming of three-dimensional components. *Bänder Bleche Rohre*, 35(11):26–30, November 1994. (in German).

[10] Manfred Geiger, Jürgen Kraus, and Frank Vollertsen. Material properties of section after laser beam bending. *Bänder Bleche Rohre*, 35(12):18–21, December 1994. (in German).

[11] Manfred Geiger, Frank Vollertsen, and Günter Deinzer. Flexible straightening of car body shells by laser forming. In *Sheet Metal and Stamping Symposium SP-944*, pages 37–44. Society of Automotive Engineers, Inc., March 1993.

[12] D.E Goldberg. *Genetic Algorithems in Search, Optimization, and Machine Learning.* Addison-Wesley Publishing Company Inc., Reaging, Massachusetts, 1989.

[13] G. N. Haidemenopolos. Heat flow and material degradation during laser metal forming. S.M. thesis, Department of Ocean Engineering, Massachusetts Institute of Technology, Cambridge, Mass., 1985.

[14] Toshiro Hashimoto. Line heating method -a new technique taking the place of Smith's work-. In *The Society of Naval Architectects of Japan 60th Anniversary Series*, number 5 in 60th Anniversary Series, chapter 3, pages 53–71. The Society of Naval Architects of Japan, Tokyo, Japan, 1 1957.

[15] E. Hinton and D.R.J. Owen. *Finite Element Programming.* The Art of Computer Programming. Academic Press, London, 1977.

[16] Y. Iwamura and E. F. Rybicki. A transient elastic-plastic thermal stress analysis of flame forming. *Transactions of ASME, Journal of Engineering for Industry*, pages 163–171, February 1973.

[17] Yasuhiro Iwasaki, Ryoji Taura, Hiroshi Shiota, and Akira Fjikura. Study on the forming of hull plate by line heating method. *Mitsubishi Juko Giho*, 12(3):51–59, 1975. (in Japanese).

[18] R. C. Johnson. Real-time sensing and control of out-of-plane distortion due to line heating by laser. S.M. thesis, Department of Ocean Engineering, Massachusetts Institute of Technology, Cambridge, Mass., 1984.

[19] Stefan Kittel and Frank Küpper. Laser beam forming of sheets with induced residual stresses. *Bänder Bleche Rohre*, 34(3):54–62, March 1993. (in German).

[20] Reiner Kopp, Stefan Kittel, and Christian Scholl. Simulation of the laser forming of steel sheets. *Bänder Bleche Rohre*, 35(10):18–21, October 1994. (in German).

[21] W. H. Luebke. *Thermo-Mechanical Plate Forming System Concepts Using Laser Line Heating.* Ph.D. thesis, Massachusetts Institute of Technology, Department of Ocean Engineering, 1987.

[22] H. A. Malaret. Mechanisms in thermal mechanical forming of plates. S.M. thesis, Department of Ocean Engineering, Massachusetts Institute of Technology, Cambridge, Mass., 1987.

[23] K. Masubuchi, G. N. Haidemonopoulos, and W. H. Luebke. Forming of steel plates by high-power laser beam. In *Abstracts of the 69th Annual Convention of the American Welding Society*, April 1988.

[24] K. Masubuchi and W. H. Luebke. Phase II (Final Report) on laser forming of steel plates for ship construction. Submitted from Massachusetts Institute of Technology, Cambridge, Mass., to Todd Pacific Shipyards Corporation, March 1987.

[25] Koichi Masubuchi. *Analysis of Welded Structures.* Pergamon international library: international series on materials science and technology; 33. Pergamon Press Ltd., Oxford, U.K., 1980.

[26] Koichi Masubuchi. Studies at M.I.T. related to laser technologies in metal fabrication. In A. Matsunawa and S. Katayama, editors, *Proceedings of an International Conference on Laser Advanced Materials Processing (LAMP92) Vol.2*, pages 939–946, Nagaoka, Japan, June 1992.

[27] Koichi Masubuchi, W. J. Cook, C. D. L. Deacon, J. S. Fan, G. Haidenmenopoulos, R. C. Johnson, and R. W. McCarthy. Phase I Report on laser forming of steel plates for ship construction. Submitted from Massachusetts Institute of Technology, Cambridge, Mass., to Todd Pacific Shipyards Corporation, February 1985.

[28] Koichi Masubuchi, I. Masumoto, and members of HPL Committee. Study on metal working by high-power laser. In *Proceedings of the Second International Conference on Beam Technology*, pages 20–24, Essen, Federal Republic of Germany, September 1985.

[29] R.M. McCarthy. Thermomechanical forming of steel plates using laser line heating. S.M. thesis, Department of Ocean Engineering, Massachusetts Institute of Technology, Cambridge, Mass., 1985.

[30] R.D. Mindlin. Influence of rotatory intertia and shear on flexural motions of isotropic elastic plates. *Journal of Applied Mechanics*, 18:31–38, 1951.

[31] A. Moshaiov and R. Latorre. Temperature distribution during plate bending by torch flame heating. *Journal of Ship Research*, 29(1):1–11, March 1985.

[32] A. Moshaiov and W S. Vorus. Elasto-plastic plate bending analysis by a boundary element method with initial plastic moments. *International Journal of Solids and Structures*, 22(11):1213–1229, November 1986.

[33] Amiram Moshaiov and Jong Gye Shin. Modified strip model for analyzing the line heating method -Part 2: Thermo-elastic-plastic plates. *Journal of Ship Research*, 35(3):266–275, September 1991.

[34] Amiram Moshaiov and William S. Vorus. The mechanics of the flame bending process: Theory and applications. *Journal of Ship Research*, 31(4):269–281, December 1987.

[35] Isao Neki, Jun-ichiro Ogawa, and Ryoichi Kamichika. A simulation analysis on the thermal elasto-plastic deformation of steel plate by high frequency induction heating. *Journal of the Society of Naval Architects of Japan*, 174:535–542, December 1993. (in Japanese).

[36] Fujio Nishioka, Masakatsu Matsuishi, Wataru Yasukawa, Osamu Tohgoh, Koh Nishimaki, Takeshi Tanaka, and Toshio Yamauchi. On automatic bending of plates by the universal press with multiple piston heads (1st Report: Fundamental study). *Journal of the Society of Naval Architects of Japan*, 132:481–501, December 1972. (in Japanese).

[37] Toshiharu Nomoto, Takuya Ohmori, Taku Sudo, Makoto Enosawa, Kazuhiro Aoyama, and Masao Saitoh. Development of simulator of plate bending by line-heating. *Journal of the Society of Naval Architects of Japan*, 168:527–535, December 1990. (in Japanese).

[38] Kevin Scully. Laser line heating. *Journal of Ship Production*, 3(4):237–246, November 1987.

[39] Jong Gye Shin. *Three-dimensional Simulation of Thermo Elastic Plastic Plate Bending by the Line Heating Method.* Ph.D. thesis, Massachusetts Institute of Technology, Department of Ocean Engineering, February 1989.

[40] Jong Gye Shin and Amiram Moshaiov. Modified strip model for analyzing the line heating method -Part 1: Elastic plates. *Journal of Ship Research*, 35(2):172–182, June 1991.

[41] M.D. Snyder and K.J. Bathe. A solution procedure for thermo-elastic-plastic and creep problems. *Journal of Nuclear Engineering and Design*, 64:49–80, 1981.

[42] Yukio Ueda, Hidekazu Murakawa, Ahmed Mohamed Rashwan, Ryoichi Kamichika, Morinobu Ishiyama, and Jun-ichiro Ogawa. Development of computer aided process planning system for plate bending by line-heating (4th Report) decision making on heating condition, location, and direction. *Journal of the Society of Naval Architects of Japan*, 174:683–695, December 1993. (in Japanese).

[43] Yukio Ueda, Hidekazu Murakawa, Ahmed Mohamed Rashwan, Ryoichi Kamichika, Morinobu Ishiyama, and Jun-ichiro Ogawa. Development of computer aided process planing system for plate bending by line-heating (Report IV) -decision making on heating condition, location, and direction-. *Transactions of JWRI*, 22(2):305–313, 1993.

[44] Yukio Ueda, Hidekazu Murakawa, Ahmed Mohamed Rashwan, Isao Neki, Rryoichi Kamichika, Morinobu Ishayama, and Jun-ichiro Ogawa. Development of computer aided process planning system for plate bending by line-heating (3rd Report) relation between heating condition and deformation. *Journal of the Society of Naval Architects of Japan*, 173:409–419, June 1993. (in Japanese).

[45] Yukio Ueda, Hidekazu Murakawa, Ahmed Mohamed Rashwan, Isao Neki, Ryoichi Kamichika, and Jun-ichiro Ogawa. Development of computer-aided process planning system for plate bending by line-heating (3rd Report) relation between heating condition and deformation. *Journal of Ship Production*, 10(4):248–257, November 1994.

[46] Yukio Ueda, Hidekazu Murakawa, Ahmed Mohamed Rashwan, Yasuhisa Okumoto, and Ryoichi Kamichika. Development of computer aided process planning system for plate bending by line-heating (1st Report) relation between the final form of the plate and the inherent strain. *Journal of the Society of Naval Architects of Japan*, 170:577–586, December 1991. (in Japanese).

[47] Yukio Ueda, Hidekazu Murakawa, Ahmed Mohamed Rashwan, Yasuhisa Okumoto, and Ryoichi Kamichika. Development of computer aided process planning

system for plate bending by line-heating (2nd Report) practice for plate bending in shipyard viewed from aspect of inherent strain. *Journal of the Society of Naval Architects of Japan*, 171:405–415, June 1992. (in Japanese).

[48] Yukio Ueda, Hidekazu Murakawa, Ahmed Mohamed Rashwan, Yasuhisa Okumoto, and Ryoichi Kamichika. Development of computer-aided process planning system for plate bending by line-heating (1st Report) -relation between the final form of the plate and the inherent strain. *Journal of Ship Production*, 10(1):59–67, February 1994.

[49] Yukio Ueda, Hidekazu Murakawa, Ahmed Mohamed Rashwan, Yasuhisa Okumoto, and Ryoichi Kamichika. Development of computer-aided process planning system for plate bending by line-heating (2nd Report) practice for plate bending in shipyard viewed from aspect of inherent strain. *Journal of Ship Production*, 10(4):239–247, November 1994.

[50] Yukio Ueda, Hidekazu Murakawa, Jianhua Wang, and Min Gang Yuan. Three dimensional numerical simulation of various thermo-mechanical processes by FEM (Report II) -deformation analysis of compressor under assembly by shrinkage fit-. *Transactions of JWRI*, 21(2):119–124, 1992.

[51] Yukio Ueda, Jianhua Wang, Hidekazu Murakawa, and Min Gang Yuan. Three dimensional numerical simulation of various thermo-mechanical processes by FEM (Report I) -methods for improving the convergence of 3-D analysis of welding-. *Transactions of JWRI*, 21(2):111–117, 1992.

[52] U.S. Department of Transportation, Maritime Administration. Line heating. The National Shipbuilding Research Program, Stanford, California, November 1982.

[53] F. Vollertsen and S. Holzer. *3D-Thermomechanical Simulation of Laser Forming*, pages 785–791. Simulation of Materials Processing: Theory, Methods and Applications. Balkema, Rotterdam, 1995.

[54] R. D. Webb and D. E. Hardt. A transfer function description of sheet metal forming for press control. *Transactions of ASME, Journal of Engineering for Industry*, 113:44–52, February 1991.

[55] O.C. Zienkiewicz and C.J. Parekh. Transient field problems: Two dimensional and three dimensional analysis of isoparametric finite elements. *International Journal for Numerical Methods in Engineering*, 2(1):61–71, 1970.