GOVP1199308104

주조금형 최적화를 위한 해석용 프로그램 개발에 관한 연구(II)

A Study on Development of a Numerical Analysis Program for the Optimization of Casting Mould (II)

연구기관 : 한국과학기술원

과 학 기 술 처

제출문

과학기술처장관 귀하

본 보고서를 "주조금형 최적화를 위한 해석용 프로그램의 개발에 관한 연구"의 2차년도 보고서로 제출합니다.

1992년 7월 21일

총괄연구책임자 : 나 석 주

연 구 원 : 노 태 정

이 승 영

배 강 열

김 원 배

김 종 민

요 약 문

1. 제 목

주조금형 최적화를 위한 해석용 프로그램의 개발에 관한 연구(II)

2. 연구개발의 목적 및 중요성

생산품 가공 방법으로써의 주조(casting)는 다른 가공공정에 비해 적은 생산 공정으로 제품을 생산할 수 있는 가공 방법중의 하나이다. 적은 생산공정에 따른 높은 생산성과 복잡한 형상의 제품을 비교적 손쉽게 제작할 수 있다는 장점때문에 많은 분야에서 주조를 이용한 제품 및 부품을 생산, 사용하고 있지만 실제 주조품 생산현장에서는 제품 품질에 매우 중요한 영향을 미치는 응고제현상에 대한 정확한 분석자료 및 구체적인 인식없이 대부분의 공정 변수들이 주로 숙련된 기술자의 경험에 의존하여 결정, 제품을 생산하고 있는 실정이다. 그러나 최근의 다품종 소량생산체제에서는 주형의 종류나 형태의 변화, 또는 제품 재질의 변화에 대한 주조공정 변수의 설정은 숙련된 기술자의 경험만으로는 한계가 있다.

따라서, 우수한 품질의 주조품을 얻기위해서는 주조공정 전체에 대한 체계적인 기술축적이 필요하며 이를 위해 수치해석을 이용한 공정해석 기술확보가 절실하게 필요하며 또한 수치해석 결과를 현장작업자들이 쉽게 이해하고 사용할 수 있는 시스템의 개발이 요구된다.

3. 연구내용, 방법 및 범위

본 연구에서는 주조공정의 유한요소 해석과 사용자 인터페이스 프로그램

기 위한 유한요소 프로그램과 이 해석 결과를 그래픽으로 나타내기 위한 사용자 인터페이스 프로그램을 개발하였다. 사용자 인터페이스 프로그램 개발을 위하여 X 윈도우 시스템과 X 라이브러리 프로그램 기술을 도입하였다. 사용자 인터페이스 프로그램에서는 마우스와 키보드를 통하여 각종 데이타를 입력하고 화면에 그 결과를 나타내며 사용자는 이를 통해 유한요소 해석에 필요한 형상 및 각종 데이타를 간단히 입력시킬 수 있으며 또한 유한요소 해석결과를 쉽게 이해할 수 있다.

4. 연구 결과 및 활용에 관한 건의

사용자 인터페이스 프로그램을 사용하여 주물공정의 온도분포를 유한요소법을 이용하여 해석하였다. 해석결과를 통해서 최후의 응고위치를 찾을 수 있었다. 즉, 그 위치에서는 기공이나 기포 존재의 가능성이 있기 때문에 금형설계시 유용한 데이타로 사용될 수 있다. 유한요소 해석과 그래픽 사용자 인테페이스 프로그램의 결합으로 주물공정해석을 효과적으로 할 수 있었으며, 특히 실제 현장에서 유용하게 사용될 수 있을 것이다.

SUMMARY

1. Title

A Study on the Development of a Numerical Program for the Optimization of Casting Mould

2. Objective

Casting is one of material processing technologies by which raw the material is converted into a final useful shape. It has a less number of the manufacturing process than others. Moreover, the product of complex shapes, of having hollow sections or internal cavities, of parts that contain irregular curved surfaces, and of parts made from metals which are difficult to machine can be produced by this process. Because of these obvious advantages, casting is considered as one of the most important manufacturing processes. But at actual manufacturing spot, in fact, most process variables are determined by only skilled technician's experience, which is resulted mainly from the lack of understanding of the solidification process. Moreover, the experience has been limited for selecting the composition of cast alloys and the type and shape of the mold. Therefore, a numerical analysis is needed for the systematic data. Also, it is required to develop a software for easily understanding and using the results of the numerical analysis.

3. Scope

The main subject of this study is to develop the finite element and user interface program for analyzing the casting process. For the development of the user interface program, X Window System and X library programming technique were introduced. The user outputs are produced according to the user's interaction. Therefore, the user can simply prepare the data which are necesary for the finite element process and easily understand the results of the finite element analysis.

4. Results and proposals

Using the data prepared from the user interface program, the casting process was analyzed and the temperature distribution was determined by the finite element method. These results can be used to predict the position of the latest solidifying region. i.e., the region having the possibility of the porosity or cavity presence and consequently the useful data for mold design.

It is considered that the approach of combining the finite element method(F.E.M) and the graphical user interface(GUI) program will be effective for analyzing the casting process, especially in the practical manufacturing site.

CONTENTS

여 백

목차

여 백

NOMENCLATURE

a : thermal diffusivity

β : bulk modulus

c : specific heat

C_1 : specific heat in solid region

C_2 : specific heat in liquid region

C_f : specific heat between solid and liquid region

[C] : heat capacity matrix

ΔT_f : difference between liquidus and solidus temperature

ε : emissivity

f : rate of internal heat generation

F : view factor

Gr : Grashof number

h : heat transfer coefficient

h_c : heat transfer coefficient of convection

h_r : heat transfer coefficient of radiation

k : thermal conductivity

[K] : heat conductivity matrix

L : latent heat

n : normal vector to boundary

Nu : Nusselt number

P : Stephan - Boltzmann constant

Pr : Prandtl number

q : heat flow rate

q_i : heat flow rate given at boundary

Ra : Rayleigh number

ρ : density

Γ_1 : boundary where specific boundary temperature is given

Γ_2 : boundary where heat flow is specified

Γ_3 : boundary where convective heat transfer occurs

$s(t)$: position of solidification front

t : time

T : temperature

T_o : initial temperature in domain

T_a : temperature of atmosphere

T_{f1} : solidus temperature

T_{f2} : liquidus temperature

T_{ref} : reference temperature

T_s : temperature of boundary where convection occurs

$\bar{T}$: given boundary temperature

ν : kinematic viscosity

μ : absolute viscosity

Ω : two dimensional domain

x, y : coordinates of two dimensional domain

I. 서론

1-1. 연구 배경

주조(casting)은 고대시대부터 사용된 금속성형(metal forming) 방법으로서 복잡하고 다양한 형상의 제품을 1차의 공정에 의하여 제조 할 수 있다는 장점때문에 많은 부분에서 응용되고 있는 기술이다. 그러나 다른 가공공정과는 달리 공정중에 재료의 상(phase)이 변화하고 대부분 복잡한 형상의 제품생산에 응용되기 때문에 균일한 품질의 제품을 얻기가 어렵고 결함의 예측 및 방지가 다른 공정에 비해 어렵기 때문에 주조제품의 품질 및 작업능률의 향상을 위해서는 적절한 주조 공정의 선택이 매우 중요한 과제로 대두되고 있다.

그러나 지금까지의 주조기술은 체계적인 해석 과정을 통한 데이타의 축적이라기 보다는 대개 작업자의 경험과 직관에 의해 진행, 발전되어 왔으며, 따라서 제품의 형상 및 소재의 변화등에는 많은 시행 착오를 통하여 적정 작업조건을 선정하는등 비능률적인 요소가 많이 존재하여 생산성의 향상을 위해서는 주조공정 전체에 대한 체계적인 연구의 필요성이 절실하게 요구되고 있는 실정이다.

이에 따라 최근 선진각국의 많은 연구자들이 수치해석법을 이용한 주조공정시 응고 및 물질유동공정의 해석등에 관심을 보이고 있으며 모울드(mold)내에서의 응고 과정중의 온도분포 해석 [1-11], 열응력해석, 용융금속의 유동현상을 시뮬레이션(simulation)한 결과들이 속속 발표가 되고 있다.[12-15] 이러한 시뮬레이션 결과를 CAD(Computer Aided Design) 시스템과 접목시켜 보다 효율적인 금형설계기술을 개발하고자 하는 시도가 일부 선진국에서 연구, 개발되고 있다.

따라서 국내 주조 공업의 활성화를 위해서는 주조공정의 전체적인 해석을 통한 적절한 주조조건의 도출 및 주조 금형설계의 최적화에 대한 프로그램의 개발이 시급히 요구되고 있는 실정이다.

1-2. 연구 목적 및 범위

주조공정은 사용하는 재료의 용융점 이상에서 상온에 이르기까지 주조품의 온도 및 조직변화가 매우 심한 공정으로 주조품 내에 나타나는 응력 및 결함의 분포도 매우 복잡한 양상을 갖게 된다. 또한 이것은 금형의 상태와도 밀접한 관계를 갖기 때문에 주조품의 특성을 예측하기 위해서는 전체적인 주조공정의 체계적인 해석을 우수한 주조품 제작에 유용한 데이타로서 활용될 수 있다. 그러나 이러한 주조공정 전체의 해석은 수치해석프로그램을 위한 데이타 준비와 그 해석결과의 이해에 많은 시간과 수고가 따르게 된다. 이러한 어려움을 극복하고 현장의 실제 사용자가 신속하고 손쉽게 주조공정해석의 결과를 인식할 수 있도록 하는 것이 중요하다.

이에 본 연구에서는 주조공정해석을 보다 효율적이며 신속하게 하기위해 실제 주조금형 도면작업이 행하여지는 CAD 프로그램에서 완성된 금형도면을 금형의 형상 및 그에 따른 기학적 정보를 담고 있는 도면교환파일(DXF file)로 만들어 이를 자동격자생성 및 유한요소법을 이용한 주조공정 온도해석 프로그램에서 이용하도록 한다. 주조공정해석 프로그램에서는 이 도면교환파일을 근거로 금형과 주조품의 물성치(material property), 자동격자생성(automatic mesh generation) 등 주조공정해석에 필요한 전처리(preprocess)과정을 사용자 인터페이스(user interface) 형식으로 행한다. 이 전처리기(preprocessor)의 결과로 주조공정을 해석하고 그 해석결과를 다시 사용자가 주조금형의 설계가 만족스러운지 확인할 수 있도록 후처

리기(postprocessor)를 통해서 이를 보여주도록 한다. 그 결과가 결함이 없는 우수한 주조제품을 생산하는 데 만족스럽지 않을 경우 금형설계도면을 변경하여 위의 과정들을 반복하도록 한다. 이러한 일련의 작업들은 운영체계(operating system)로 UNIX를 사용하는 워크스테이션(workstaion)의 X 윈도우 시스템[16]하에서 행하여지며 Fig. 1에 그 작업흐름도를 나타내었다.

이러한 주조금형해석 프로그램을 이용하여 현장의 작업자에 의존한 경험적인 주조방안이 아니라 정량적인 주조공정변수 조절이 가능하도록 하여 신속한 주조제품의 개발과 생산에 그 가능성을 부여하고자 하는 것이 본 연구의 목적이다.

II. 본 론

2-1. X 윈도우 시스템(X Window System) 환경 및 프로그래밍[16]

2-1-1. X 윈도우시스템의 개요 및 역사

초기 컴퓨터가 보급되기 시작할 무렵의 컴퓨터관련 작업은 프로그램밍 코드 전체를 펀치카드(punch card)를 입력하고 결과를 기다리는 일련의 배치(batch)작업형식으로 진행되었다. 그 후 비디오 디스플레이 터미날(video display terminal), 미니 컴퓨터(mini computer), 인터액티브 운영체계(interactive operating system)가 도입됨으로해서 프로그램을 실행시키기가 용이하여졌으며 배치(batch)작업으로 인한 시간적 손실등의 여러가지 어려움을 극복하기에 이르렀다.

더욱이 최근에 들어서는 퍼스널컴퓨터(personal computer)와 워크스테이션(workstation)에 그래픽 사용자 인터페이스(Graphic User Interface, GUI)가 도입되므로해서 새로운 전기를 맞게 되었다. 그래픽 사용자 인터페이스(GUI)는 종래의 키보드에 의한 컴퓨터와의 인터액션(interaction)에서 발전하여 대부분의 작업들을 포인팅시스템(pointing system), 즉 마우스(mouse)를 이용하여 파일의 편집, 복사, 제거등의 작업이 이루어질 수 있도록 하였다.

또한 하나의 프로세서에 대한 출력만이 가능한 종래의 텍스트(text)전용 터미날에서 하나의 윈도우가 가상의 실질적 터미날 역활을 하는 다중 윈도우의 구현이 가능한 윈도우시스템(window system)과 이를 실현시킬 수 있는 그래픽터미날(graphic terminal)로 대체하므로써 출력과 계산의 분리, 시각적 효과 등으로 컴퓨터작업을 신속하고 효율적으로 수행할 수 있도록 하였다.

최근 컴퓨터관련작업의 대용량, 고처리속도 요구에 부합하고 기술개발에 따른 컴퓨터 가격인하에 상응하여 워크스테이션급 컴퓨터의 사용이 급속한 신장세를 나타내고 있다. 이 워크스테이션 컴퓨터들은 퍼스날컴퓨터들와 달리 다양한 회사에서 호환성을 갖지 않는 하드웨어와 다양한 운영체계(operating system)로 생산되고 있다. 이 워크스테이션들은 멀티테스킹(multi-tasking)기능과 네트워크(network)기능으로 업무처리의 효율성을 높이고 있다. 그러나 기종이 다른 제품의 워크스테이션들은 상이한 하드웨어 구조와 운영체계, 프로그램 개발에 사용되는 라이브러리(library)나 라이브러리 실행 루틴 등이 다르므로 워크스테이션 상호간의 프로그램의 호환성이 전혀 없는 실정이다.

X 윈도우 시스템은 서로 다른 기종의 하드웨어에서도 동일한 프로그램 수행이 가능하도록 사용자에게 프로그램의 하드웨어에 대한 독립성을 부여하고 또한 멀티테스킹과 네트워크기능을 부여하며 그래픽 디스플레이 터미날을 채택하여 다중의 윈도우시스템을 지원, 관리하므로써 효율적 터미날 사용으로 프로그램개발에 효율성을 부여하고 있다.

더욱이 X 윈도우 시스템은 터미날상의 윈도우관리 유틸리티(utility)와 강력한 C언어의 라이브러리를 지원하므로써 사용자에게 그래픽구현 등 프로그램 개발의 편리함을 더하여 사용자 그래픽 인터페이스(GUI) 구현을 용이하게 한다.

Fig.2는 X 윈도우 시스템을 채택한 워크스테이션을 나타내었다. 이러한 X 윈도우 시스템은 위에서 언급한 필요성으로 1984년 DEC사와 IBM사의 참여속에서 MIT에서 개발되기 시작해으며 1986년도에 울트릭스(Ultrix) 운영체계의 VAXstation-II/GPX에서 운용되는 사업용으로 처음 선을 보였다. 그 후 휴렛패커드(Hewlett Packard)사, 아폴로 컴퓨터(Apollo Computer), 선 마

이크로시스템(Sun Microsystems), 텍트로닉스(Tektronix)등의 관심을 끌기 시작하였고 그 동안 1986년 X10R3에서 출발하여 현재 1990년의 X11R4에 이르고 있다. 현재 그래픽터미날을 사용하는 워크스테이션은 대부분 이 X 윈도우 시스템을 채택하고 있다.

2-1-2. X 윈도우 시스템의 구성

X 윈도우 시스템은 응용(application) 프로그램인 클라이언트(Client)와 이 클라이언트(Client) 프로그램의 실행, 디스플레이의 관리, 키보드와 마우스의 입출력 관리, 네트워킹되어 있는 단말기에서의 출력관리 등을 맡는 서버(Server), 클라이언트와 서버를 연결시켜 주고 받는 데이타들를 해석하는 프로토콜(Protocol), 그리고 프로그램개발에 사용되는 X 라이브러리 루틴(library routine)으로 이루어져 있다. 즉 X 윈도우 시스템은 클라이어트 어프리케이션(Client application), 서버(Server), 프로토콜(Protocol), X 라이브러리(X library)를 총칭하는 것을 의미한다.

X 서버는 워크스테이션내의 프로세서를 관리하고 그래픽 출력과 키보드와 마우스의 입력을 관리하는 디스플레이 서버(display server)와 네트워킹된 단말기를 관리하는 파일 서버(file server) 그리고 데이타베이스 서버(database server)로 구성되어 있으며 이중 가장 중요한 것은 X 디스플레이 서버(X display srever)이다. X 서버는 X 윈도우 시스템의 근간으로 다중의 윈도우로 구성되어 있는 디스플레이 터미날에서 하나의 윈도우가 가상의 실제적인 디스플레이 터미날의 역활을 하는 X 윈도우 시스템에서 각각의 윈도우에서 발생하는 이벤트(event)들을 각 윈도우를 소유한 클라이언트들에게 알려주고 프로세서를 관리하는 역활을 한다.

X 프로토콜은 X 클라이언트가 X 서버와의 정보교환에 사용하는 데 이 X

프로토콜는 마이크로프로세서에 비유하면 직접 시스템에 작용하는 X 윈도우 시스템의 기계어(machine language)라고 볼 수 있다.

X 라이브러리은 그 자체가 C언어의 특별한 라이브러리와 함수, 데이터 구조체이다. X 서버와 사용자가 직접 X 프로토콜을 사용하여 정보를 교환하는 것은 매우 어려운 일이다. X 프로토콜을 기계어에 비유하는 것도 이러한 점을 감안한 것이다. X 라이브러리는 마이크로프로세서에 비유하면 어셈블리언어(assembly language)할 수 있다.

그외 X 라이브러리에 비해 더욱 고수준(high-level) 언어 역활을 하는 X 툴키트 인트린식스(X Toolkit Intrinsics)이 있으며 이는 사용자인터페이서에 용이한 도구를 제공하여 주는 역활을 한다. 그러나 X 윈도우 시스템을 이용한 프로그래밍에는 X 라이브러리를 사용하는 것은 필수적이다. Fig.3에 X 적용의 일반적인 구조를 나타내었다.

2.1.3 X 라이브러리 프로그래밍

X 윈도우 시스템은 클라이언트-서버 모델을 사용한다. X 서버는 디스플레이에서의 입력과 출력을 관리하여 클라이언트의 요구에 상응한 프로세서를 수행하고 그 결과 및 클라이언트가 사용하는 윈도우에서 발생하는 모든 이벤트를 클라이언트에 알려주게 된다. 이러한 X 클라이언트들은 X 서버와의 정보교환을 X 프로토콜 요구(protocol request)를 통하여 하게 되며 이러한 X 서버를 사용자가 직접 X 프로토콜을 통하여 프로그램한다는 것은 기계어로 복잡한 프로그램을 개발하는 것과 같다.

이에 X 윈도우 시스템은 사용자가 X 서버에게 X 프로토콜 요구를 보낼 수 있도록 C언어 함수들과 맥크로(macro)들로 이루어진 X 라이브러리를 제공하고 있다. 즉 X 라이브러리의 목적은 X 서버의 특성을 활용할 수 있는

X 프로토콜 요구를 X 서버에게 보내는 것이다. 또한 X 라이브러리 프로그래밍 모델은 각 크라이언트의 요구하는 프로세서를 X 서버가 수행하고 그 각각의 영역에서 발생하는 각종 이벤트를 알려주는 루틴으로 수행되므로 이벤트 주도(event-driven) 모델이라고 불리운다. Fig. 4에 일반적인 이벤트 주도 프로그램를 나타내었다.

Fig.4에서 초기화(initialization) 단계는 응용 프로그램이 사용자로부터 입력을 받거나 출력을 디스플레이할 경우 사용되는 리소스(resouce)들을 X 서버에 만드는 과정이다. X 서버에 있어서 리소스(resource)란 윈도우(window), 폰트(font), 그래픽 콘택스트(graphic context), 픽스맵(pixmap), 컬러맵(colormap) 등을 말한다. 초기화은 다음과 같은 과정을 거치게 된다.

· 현재 사용하고자 하는 디스프레이와의 연결

· 사용자가 원하는 폰트들과 컬러들의 선택

· 프로그램이 사용할 최상위 윈도우(top-level window)의 생성

· 윈도우의 속성을 윈도우 매니저(window manager)에 알림

· 윈도우에 사용될 그래픽 콘택스트(graphic context)의 생성

· 윈도우에서 수행될 이벤트의 선정

· 윈도우를 화면에 출현

위의 초기화을 마치고 나서 실제로 X 라이브러리 프로그래밍에서 가장 중요한 이벤트 루프(event loop)를 수행하게 되며 이 이벤트 루프에서 수행하게 될 이벤트는 초기화단계에서 선택된 이벤트들이다.

해제(clean-Up) 단계는 프로그램을 끝내기 앞서 X 서버에 만들어진 리소스들이 차지하고 있던 기억장소를 비우는 작업이다.

다음은 본 연구에 사용된 X 라이브러리 프로그램중 주요한 부분에 대

해 소개를 하였다.

· 윈도우(window)

윈도우의 활용은 X 윈도우 시스템을 사용하는 가장 큰 목적 중의 하나이다. 윈도우는 출력을 디스플레이하고 입력을 받아 들며 그래픽을 출력하거나 이벤트 처리(event handling) 등을 수행할 경우 반드시 관련된 윈도우를 지정해야 한다. X 윈도우 시스템의 윈도우 구조는 부모-자식(parent-child) 관계를 갖는 계층적 구조(hierarchical structure)이며 자식 윈도우는 그 상위 윈도우의 속성을 받는다.

본 연구에 사용된 윈도우체계는 최상위 윈도우를 부모 윈도우로 하여 여러개의 자식 윈도우들를 만들고 자식 윈도우들을 통해 그래픽의 출력, 데이타의 입력을 받았다.

· 이벤트 처리(event handling)

X 라이브러리 프로그래밍과정 중 실제 수행하고자 하는 프로세서를 처리하는 과정으로 각 윈도우에 대해 각각 그 윈도우에서 다루고자 하는 이벤트들을 윈도우를 화면상에 나타내기 전에 윈도우 속성(window attribute)으로 지정하여 주어야 한다. 실제 어떤 이벤트들을 특정 윈도우에 지정하는 프로그램의 일부를 아래에 나타내었다.

```
Display *p_dis;
Window  *Main;

XSelectInput(p_disp,Main,ButtonPressMask¦KeyPressMask);
```

위 프로그램의 뜻은 X 서버와 이름 p_disp로 연결된 터미날의 윈도우 Main에게 마우스버턴입력 이벤트과 키보드입력 이벤트를 지정하는 것이다.

이렇게 지정된 이벤트들은 X 라이브러리 함수인 XNextEvent(p_disp, &theEvent)를 사용하여 X 서버로부터 이벤트 정보를 저장하는 데이타구조를 통하여 받아들이게 된다. 본 연구에서는 이러한 이벤트 처리을 통해 원하는 입력과 출력을 할 수 있었다.

· 그래픽스(graphics)

본 연구에서는 주조공정해석 결과인 온도분포를 해석영역에 색깔로 표현하고자 하였다. X 라이브러리 프로그래밍에서 그래픽을 구현하고자 하면 먼저 주어진 픽셀(pixel)값에 대해 하드웨어가 구현할 수 있는 가장 적합한 색깔로 변환시켜주는 컬러맵(colormap)을 먼저 지정해 주어야 하며 이 colormap에서의 얻어진 색깔들로 graphic context들을 지정해 주었다. 각각의 온도영역에 대해 하나씩의 그래픽 콘택스트(graphic context)를 정의하였고 해석영역의 온도가 지정된 특정범위의 온도이면 그 범위에 지정된 그래픽 콘택스트로 그 영역의 색깔을 결정하였다.

이상의 X 라이브러리 프로그래밍의 흐름도를 Fig.5에 나타내었다.

2.2 자동격자생성

2-2-1. CAD프로그램의 도면교환파일

결함없는 우수한 품질의 주조제품을 생산하기 위해서는 주조금형의 설계형태가 큰 부분을 차지한다. 실제로 주조금형의 설계는 범용의 CAD프로그램에서 이루어지는데 이 CAD프로그램에서 만들어진 금형의 설계도면을 곧바로 주조공정해석 프로그램에 이용할 수 있도록 하고 이 해석 결과를 토대로 다시 금형의 설계를 수정하므로써 효율적인 주조품을 생산할 것으로 기대된다.

범용 CAD프로그램에서의 금형설계도면이 주조공정해석 프로그램에 이용될 때에는 설계된 금형의 기하학적 정보가 도면교환파일(DXF File) 형태로 만들어져 사용된다. Fig.6는 범용 CAD프로그램 중의 하나인 AutoCAD에서 작성된 도면과의 그 기하학적 정보를 가진 도면교환파일을 나타내었다. 퍼스날 컴퓨터용 혹은 워크스테이션용 CAD프로그램은 그 프로그램자체에 대부분 도면교환파일을 생성하는 기능들을 가지고 있다.

2-2-2. 자동격자생성 알고리즘

유한요소법에 의한 수치해석에는 해석하고자 하는 전체영역의 미소영역으로의 분할과 그에 따르는 각종 정보를 사전에 준비하여 입력해 주어야만 한다. 그러나 이러한 유한요소해석에 필요한 정보를 준비하는 전처리(preprocess)는 그 준비해야 할 데이타의 양이 엄청날 뿐만 아니라 사용자가 그 데이타를 직접 입력한다는 것은 매우 번거럽고 시간을 많이 소요되는 작업이다.

이러한 문제점을 해결하기 위해서 유한요소법이 도입된 초기에서부터 자동격자생성 및 그에 정보를 마련하기 위해 많은 알고리즘들이 개발되어 왔다.

본 연구에서는 금형설계도면에서 생성된 도면교환파일을 받아 들여 이를 자동격자생성과 그에 따른 유한요소해석을 위한 정보 등을 마련할수 있도록 사용자 인터액션에 의한 전처리기(preprocessor)에서 이를 수행하고자 한다 (부록 1 참조). 자동격자생성 알고리즘은 아래와 같다.

· 블럭 근간 알고리즘(Block-based algorithm)[17,18]

해석하고자하는 영역을 분할이 용이한 블럭들로 나누어 이 각각의 블럭

에 대해 자동격자생성을 시키고 격자가 생성된 블럭들을 서로 연결하므로서 전체영역에 대한 격자를 생성시키는 알고리즘이다. Fig.7는 이 알고리즘에 대한 흐름도를 나타낸다. 또한 Fig.8, Fig.9는 CAD프로그램에서 작성된 도면의 도면교환파일에서 블럭 번호매김(Block numbering), 블럭 좌표와 블럭 연결(Block connecticity)를 자동생성시킨 것을 나타낸것으로서 CAD 프로그램에서 해석하고자하는 영역을 블럭으로 표현할때 블럭의 배치 및 갯수에 제약을 두지않고 임의로 그릴 수 있는 예를 나타낸것이다. Fig.8과 Fig.9는 똑 같은 영역을 나타낼경우 블럭의 수와 배치를 달리한것이다(부록2 참조).

2-3. 주조공정해석의 이론적 배경

금속응고의 제 현상은 매우 복잡하여 간단한 수식으로 표현하기 어렵다. 유체상태의 금속은 비뉴턴(Non Newtonian)유체의 성질을 가지고 있으며 이때의 열전달은 복합적인 함수관계로 진행되므로 지배방정식이 비선형이 되어 복잡한 금속응고 상태는 해석적 해법으로는 구할 수 없다.

근간에 수치해석법의 개발로 많은 공학자들이 금속응고상태를 수치적으로 해석하는 연구를 해왔다. 그러나 아직도 해결하기 힘든 난제는 정확한 지배방정식의 설정이 되어 있지 않고, 경계조건에 대한 정확한 자료의 부족이라는 점이다.

본 연구에서는 수치해석 기법중 유한요소법을 이용하여 주조공정을 모델링 하였으며 문제의 복잡성을 피하기 위하여 2차원 해법으로 해석하였다.

2-3-1. 금속응고에 대한 지배방정식

주어진 해석대상을 무한히 긴 물체라 가정한 2차원 영역 Ω에서의 비정상 열전달에 대한 지배방정식(Governing equation)은 다음과 같이 나타내진

다.

$$\rho c \left(\frac{\partial T}{\partial t} \right) - \nabla \cdot (k \nabla T) = f, \qquad x, y \in \Omega \subset R^2 \tag{1}$$

여기에서 ρ는 밀도(density)

c는 비열(specific heat)

k는 열전도율(thermal conductivity)

f는 내부열발생율(rate of internal heat generation)

T는 온도(temperature)

t는 시간(time)

Ω는 2차원 영역

x, y는 2차원영역의 좌표

이다. 이때 금형 내에서의 대류효과는 무시하였다. (1)식의 해를 구하기 위한 해석영역 내부의 초기조건은 (2)식과 같으며, 경계조건은 각각의 경계상태에 따라 (3)식과 같이 나타낼 수 있다.

i) 초기조건 $$T(x, y, 0) = T_0(x, y) \tag{2}$$

ii) 경계조건

1) 경계온도를 아는 경우

$$T(x, y, t) = \bar{T} \qquad \text{at} \quad \Gamma_1 \tag{3.1}$$

2) 열 유입이 있는 경우

$$q = \left(- k \frac{\partial T}{\partial n} \right) = q_i \qquad \text{at} \qquad T_2 \qquad (3.2)$$

3) 대류에 의한 열전달의 경우

$$q = \left(- k \frac{\partial T}{\partial n} \right) = h(T_s - T_a) \qquad \text{at} \quad T_3 \qquad (3.3)$$

여기서 To는 영역내의 초기온도

$\bar{T}$ 는 주어진 경계온도

T_1는 경계온도가 주어진 경계면

q는 열유동(heat flow rate)

q_i는 경계면에 주어진 열유동

n은 경계면에 수직인 벡터

T_2는 열유입과 열유출이 주어진 경계면

h는 대류열전달계수

T_s는 경계면의 온도

T_a는 경계면 외부의 온도

T_3는 대류열전달이 주어진 경계면

이다. 여기서 응고과정해석에 이용하기 위해서는 지배방정식을 고체(solid) 영역과 액체(Liquid) 영역에 각각 적용시켜야 하며 이때에는 위의 조건들 외에 응고표면에서 다음의 조건이 첨가되어야 한다.

$$T_1 = T_2 = T_f \qquad \text{for } t > 0 \text{ and } y = s(t) \qquad (4)$$

$$k_1\left(\frac{\partial T}{\partial n}\right)\Big|_{x=S(t)} + k_2\left(\frac{\partial T}{\partial n}\right)\Big|_{x=S(t)} = \rho L\left(\frac{\partial s}{\partial t}\right) \qquad (5)$$

여기서 L은 잠열(latent heat)

s(t)는 응고면의 위치(the position of the solidification front)

첨자 1은 고상(solid phase)

첨자 2는 액상(liquid phase)

을 나타낸다. 응고구간에서 상 변화에 기인하는 잠열(latent heat)의 방출을 고려하기 위해 고정격자 엔탈피법 (Fixed Grid Enthalpy Method)[3,4,5,10]를 사용하였다. 고정격자 엔탈피법은 응고가 진행됨에 따라 응고영역이 확대될 때 이미 응고가 완료된 고체영역과 아직 용융상태의 액체영역을 해석하는 데 있어 응고양상변화에 따른 격자의 재생성 없이 고정된 격자를 사용하고 상변화 등에서 나타나는 물성치의 급격한 변화 및 잠열의 처리를 엔탈피(enthalpy)의 도입으로 처리하는 방법이다. 용융된 합금이 유한한 범위의 응고온도 구간을 갖는다고 가정하면 엔탈피는 아래와 같이 제안된 식으로 정의할 수 있다.

응고 구간을 (T_{f1}, T_{f2}) 라 하면 이때 각 온도 구간에서의 엔탈피는

$$H = \int_{T_{ref}}^{T} \rho C_1(T)dT \qquad \text{for} \quad T < T_{f1} \qquad (6.1)$$

$$H = \int_{T_{ref}}^{T} \rho C_1(T)dT + \int_{T_{f1}}^{T}\left[\rho\left(\frac{dL}{dt}\right) + \rho C_f(L)\right]dT \quad \text{for } T_{f1}<T<T_{f2} \quad (6.2)$$

$$H = \int_{T_{ref}}^{T_{f1}} \rho C_1(T)dT + \int_{T_{f1}}^{T_{f2}} \rho C_f(T)dT + \int_{T_{f2}}^{T} \rho C_2(T)dT \quad \text{for} \quad T > T_{f2} \quad (6.3)$$

여기에서 T_{ref}는 기준점 온도(reference temperature)

T_{f1}는 고상선 온도(solidus temperature ($= T_f - \Delta T_f$))

T_{f2}는 액상선 온도(liquidus temperature ($= T_f - \Delta T_f$))

ΔT_f는 ($T_{f2} - T_{f1}$)/2

C_1는 고상영역($T_{ref} < T < T_{f1}$)의 비열

C_2는 액상영역($T_{f2} < T$)의 비열

C_f는 $T_{f1} < T < T_{f2}$ 영역의 비열

이다. 위식을 사용하여 열전달 지배방정식을 나타내면 다음과 같다.

$$C^* \left(\frac{\partial T}{\partial t} \right) - \nabla(k\nabla T) = f \quad (7)$$

여기에서 $C^* = (dH / dT)$는 유효비열(effective heat capacity)이며 구간별 유효비열의 형태는 아래와 같다.

$$\text{i)} \quad C^* = \rho C_1 \quad \text{for} \quad T < T_{f1} \quad (8.1)$$

$$\text{ii)} \quad C^* = \rho C_f + L / (T_{f2} - T_{f1}) \quad \text{for} \quad T_{f1} < T < T_{f2} \quad (8.2)$$

$$\text{iii)} \quad C^* = \rho C_2 \quad \text{for} \quad T > T_{f2} \quad (8.3)$$

또한 Del Guidice, Lemmon등에 의해 제안된 유효비열 C^* 는 다음과 같다.[10]

Del Guidice에 의하면

$$C^* = \frac{[(\partial H/\partial y)\cdot(\partial T/\partial y) + (\partial H/\partial z)\cdot(\partial T/\partial z)]}{[(\partial T/\partial y)^2 + (\partial T/\partial z)^2]} \quad (9)$$

이며

Lemmon에 의하면

$$C^* = \left\{ \frac{[(\partial H/\partial y)^2 + (\partial H/\partial z)^2]}{[(\partial T/\partial y)^2 + (\partial T/\partial z)^2]} \right\}^{1/2} \quad (10)$$

이다. 따라서 잠열을 고려한 응고과정의 열전달 지배방정식 (8)식을 Galerkin 방법을 적용하여 유한요소 수식화하여 영역(domain)에 대해 적분하면 다음과 같은 최종의 유한 요소 방정식을 얻을 수 있다.

$$\int_\Omega C^* \left(\frac{\partial T}{\partial t}\right) T d\Omega + \int_\Omega (k\nabla T)\cdot\nabla T d\Omega - \int_\Gamma (k\nabla T)\cdot n T d\Gamma = \int_\Omega f T d\Omega \quad (11)$$

$$[\ C(T)\ ]\{\ T\ \} + [\ K(T)\ ]\{\ T\ \} = \{\ F\ \} \quad (12)$$

여기에서 [C]는 열용량 매트릭스(heat capacity matrix)

[K]는 열전도 매트릭스(Heat conductivity matrix)

이다.

2-3-2. 계면에서의 열전달계수

2-3-2-1. 대기와 접한 모울드의 냉각

(1) 자연대류에 의한 경우[19]

모울드벽 가까이서 가열되어 온도가 올라간 공기는 위로 상승하게 되며

모울드 위로 올라갈수록 밑에서 상승하는 더운 공기들이 모여서 경계층도 두꺼워지고 유체의 흐름은 난류가 된다. 층류에서 난류로 바뀌는 지점은 통상 Ra가 10이 되는 곳으로 열전달해석은 층류와 난류로 구분해서 함으로 용이해진다. 열전달해석에 사용되는 중요기호는 아래와 같다.

Ra는 Rayleigh수(Ra=Gr·Pr)

Gr는 Grashof수

$$Gr = g\beta(T_s - T)/\nu^2$$

Pr는 Prandtl수($Pr = \nu/a$)

ν 는 동점성계수(kinematic viscosity)

$$\nu = \mu/\rho \ [m^2/s = \text{centistoke}]$$

μ는 절대점성계수(absolute viscosity)

$[N\cdot s/m^2 = 10^3 \text{ centipoise}]$

a는 열확산계수(thermal diffusivity)

$$a = k/\rho c \ [m^2/s]$$

β는 부피팽창계수

Nu는 Nusselt수($Nu = hx/k$)

이다. 또한 공기와 물의 중요 물성치를 표 1에 나타내었다. 모울드 각점에서 단위면적당 공기로 전달되는 열량 q 는

$$q = h(T_s - T_a) \qquad (13)$$

로 표시할때 h는 열전달계수가 되며 그점에서 유체의 성질과 유동상태에 따라 달라진다. 여기서 T_a는 모울드벽 외부온도이다. Ra수가 10^4과 10^9사이에

서는

$$Nu = hx/k = F1(Pr)(Gr/4) \qquad (14.1)$$

이고 x=0에서 x=L까지의 평균치에 대해서는

$$\bar{Nu} = \bar{hL}/k = \frac{4}{3} F1(Pr)(Gr/4)^{1/1} \qquad (14.2)$$

여기서 여러가지 물성치는 $T_{ref} = T_s - 0.38(T_s-T_a)$ 온도에서 구한 것이고 F1(Pr) 값은 표 2 와 같다

위의 해석에서는 순전히 공기의 대류에 의한 열전달만 고려한 것이고 실제에는 복사에 의한 열전달도 고려해야 한다. 따라서

$$h_t = h_c + h_r \qquad (15)$$

h_r는 열복사에 의해서 전달되는 열량을 온도차로 나눈 것으로

$$h_r = \varepsilon P F (T^2_s + T^2_a)(T_s + T_a) \qquad (16)$$

여기서

h_c는 대류에 의한 열전달계수

h_r은 복사에 의한 열전달계수

ε은 전방사율

P는 Stephan - Boltzmann상수 (5.669×10 W/m^{2}K^4)

F는 형태계수

이다. 여러가지 위치의 경계면과 유동상태에 대한 h_c값은 표 3와 같다.

2-3-2-2. 캐스트(cast)와 모울드(mold) 사이에서의 열전달계수

두 경계면이 가까이 있는 경우 이 두계면사이의 단위면적당 열전달량은

$$q = h (T1 - T2) \qquad (17)$$

여기서 T1과 T2는 두 경계면의 온도이다.

이 경우의 Grashof수는

$$Gr = g\beta(T1-T2)b^3/\nu^2 \qquad T1 \rangle T2 \qquad (18)$$

여기서 b는 양면사이의 간격이고 양면사이의 유체의 물성치는 (T1+T2)/2 온도에 해당하는 값이다.[19)]

(1) 수평 공기층(horizontal air layers)

(a) 상면이 더 고온일 경우

대류작용이 없으므로

$$q = h(T1-T2) = k(T1-T2)/b \quad 즉\ h = k/b \qquad (19)$$

또는 $Nu = hb/k = 1.0$

(b) 하면이 더 고온일 경우

Grashof수가 작을 경우 ($\langle 2000$) 대류현상은 무시할수 있고

따라서 위 식을 적용할 수 있다.

Gr가 커짐에 따라

$$Nu = (0.195)\ Gr^{1/4} \qquad 10^4 \langle Gr \langle 4\times10^5 \qquad (20.1)$$

$$Nu = (0.068)\ Gr^{1/3} \qquad 4\times10^5 \langle Gr \qquad (20.2)$$

(2) 수평 액체층(horizontal liquid layers)

수은, 물, 실리콘 오일(silicone oil)에 대해서

Pr가 0.02~8750 사이에 있을때

$$Nu = (0.069)\, Gr^{1/3}\, Pr^{0.407} \qquad 3\times10^5 < Gr\,Pr < 7\times10^9 \quad (21)$$

(3) Vertical Air Layers - isothermal walls

계면사이의 온도차이, 간격 및 유체의 성질에 따라

$$Nu = 1 \qquad Gr < 2006 \quad (22.1)$$

$$Nu = (0.18)\, Gr^{1/4}\, (L/b)^{-1/9} \qquad 2\times10^4 < Gr < 2\times10^5 \quad (22.2)$$

$$Nu = (0.065)\, Gr^{1/3}\, (L/b)^{-1/9} \qquad 2\times10^5 < Gr < 1.1\times10^7 \quad (22.3)$$

여기서 L/b > 3 이어야 한다. 이보다 간격이 클 경우에는 두개의 독립적인 수직면으로 간주하면 된다.

그러나 주괴의 열수축과 팽창에 따라 주괴-주형간에 간격을 예측하기 어려운 간격(gap)이 생성되고, 발생한 가스로 이 간격이 채워지므로 계면에서의 열전달계수는 실제로 계산하기가 매우 곤란하다.[20,21]

2-3-3. 수치 해석

유한요소법이란 미분방정식으로 표시되는 지배방정식을 영역(domain)에 대한 적분방정식의 형태로 바꾸어 이를 부영역(subdomain)에서 임의의 형상함수로 표시하여 유한요소의 집합으로 영역(domain)에서의 지배방정식을 만족하는 해를 구하는 근사해법이다. Fig.10에 본 연구에 사용된 유한요소해석의 흐름도를 나타내었다. 본 연구에서는 각 변수들을 온도의 함수로 나타내어 물질특성을 time step마다 온도의 함수로 구하고 경계면에서의 열전달계수를 인위적으로 수치화하여 해석하였다.

2-3-4. 해석 대상

본 보고서에서는 주철금형(cast iron mould)에서 알루미늄(aluminium)의 응고과정을 대상으로 하여 수치해석을 시행하였다. 전체영역을 모울드 영역, 캐스트영역, 그리고 모울드와 캐스트 사이의 공기간격(air gap)을 통한 열전달을 처리하기 위해 경계부(interface)로 나누어 해석하였다.

2-3-5. 해석 조건

물질특성을 온도에 따라 변하게 되므로 열전도율, 비열, 밀도 등을 온도의 함수로 설정하여야 한다. 주철의 함수식[11,22,23]은 다음과 같다.

$$\rho = \text{constant} = 7.85 \tag{23}$$

$$c = 0.0750\times[8.873+1.474\times10^{-3}\times(T+273)-56.92\times T^{-1/2}] \tag{24}$$

$$k = 0.385 \tag{25}$$

알루미늄[11,22,23]은 액상에서 고상으로의 상(phase)변화가 일어나므로 잠열을 처리하여야 하고 또한 열전도율도 큰 변화를 가져온다.

$$\rho = \text{constant} = 2.70 \tag{26}$$

$$c = 0.1552\times[4.94+2.96\times10^{-3}\times(T+273)] \qquad T \le 660 \tag{27.1}$$

$$= 1.1953-(1.1953-1.0864)\times(T-660)/30 \qquad 660 \le T \le 690 \tag{27.2}$$

$$= 0.1552\times7.0 \qquad 690 \le T \tag{27.3}$$

$$L = \text{constant} = 388.7$$

$$k = 2.20+0.3\times10^{-5}\times T^2 \qquad T \le 400 \qquad (28.1)$$

$$= 2.5 \qquad 400 \le T \le 660 \qquad (28.2)$$

$$= 0.94+0.000335\times(T-660) \qquad 660 \le T \qquad (28.3)$$

주철과 알루미늄의 물성치는 Fig.11에 나타내었다.

한편 본 연구의 수치해석에서는 금형외부의 열전달계수 h 를 형상을 고려하여 상수로 가정하여 해석하였다.

$$h = 0.020935 \quad [\ W/cm^2\cdot K\ ] \qquad (29)$$

또한 금형과 주괴 사이의 열전달계수에 대해서도 정확한 값을 얻기는 매우 곤란하다. 응고가 일어나면서 발생하는 주형과 주괴 사이의 공기간격(air gap)으로 인하여 열전달계수가 작아지게 되는 데 정확한 공기간격의 형태나 크기, 그에 따른 열전달계수의 값은 얻을 수 없어 공기간격의 경계부(interface)처리는 모울드과 캐스트가 경계부와 접하는 부분을 line element로 처리하였다.

초기조건은 주철금형은 예열된 상태로 보고 300°C의 초기온도를 주었고 알루미늄은 주입온도를 800°C로 주었으며 용탕이 주형내로 유입되는 과정은 고려하지 않고 일시에 주형내의 공간(cavity)를 채우는 것으로 가정하고 해석하였다.

III. 해석 결과 및 고찰

주조공정시 금형의 형상 및 경계조건의 변화에 따른 응고현상을 알아보기 위해 사용자 인터페이스 프로그램 및 유한요소 프로그램을 개발하여 금형내에서의 시간 변화에 따른 용탕의 응고양상을 해석하고, 그래픽으로 나타내었다.

Fig.12은 사용자 인터페이스 프로그램과 유한요소해석 프로그램 사이의 관계를 나타내주는 그림으로서 사용자 인터페이스 프로그램에서의 전처리과정을 통해 캐스트영역과 모울드영역의 CAD도면들의 도면교환파일을 읽어 유한요소 프로그램을 위한 격자 생성 데이터를 생성하는 과정을 보여 주고 있다. 해석 영역내의 각 블럭에 대한 블럭분할정도와 모울드영역, 캐스트 영역의 재료물성치들을 마우스와 키보드를 이용하여 입력시키는 것을 볼 수 있다.

Fig.13은 Fig.12과정에서 사용자 인터페이스 프로그램을 통해 생성된 해석영역내의 격자의 디스플레이와 유한요소해석에 필요한 데이터 처리과정을 보여주고 있다. 먼저 유한요소해석에 소요되는 시간간격(time step), 각 영역의 초기온도(initial temperature)를 받아 들인다. 또한 모울드 외부나 내부에 가하여지는 경계조건들과 그 데이타를 마우스를 이용하여 받아드리고, 특히 모울드와 캐스트의 경계부(interface)에 대한 데이타도 역시 마우스를 이용하여 받아 들인다. 격자생성의 결과가 Fig.14에 나타나 있다. 이러한 과정을 통해 형성된 격자데이터를 바탕으로 유한요소 프로그램을 통해 금형내에서의 용탕의 응고과정을 해석한다. 본 연구에서는 위에서 개발된 각 프로그램을 금형및 제품의 형상을 변화시킨 3가지의 모델에 적용하여 온

도분포 해석을 행하였다.

Fig.15는 용탕이 금형 내부에 완전히 둘러싸인 경우에 대한 온도 해석 결과로서 용탕주입후 시간 t = 1 초, t = 11 초, t = 21 초, t = 31 초일 때의 온도분포 등온선을 나타낸 것이다. t = 1 초일때의 온도분포를 보면 모울드와 캐스트 경계부 사이엔 큰 온도격차가 있는 것을 알 수 있다. 이후 시간이 지날수록 냉각이 진행되어 하단의 플랜지(flange)부분부터 응고가 일어남을 알 수 있다. 결국 시간이 경과하여 t = 11 초, t = 21 초일때 캐스트의 중심 부분에서 최후의 응고가 일어나게 됨을 알 수 있다. 이 최후에 응고되는 부분에서 응고에 따른 수축에 의한 수축공(shrinkage)나 기포, 그리고 불순물의 편석등이 발생할 가능성이 있을 것으로 생각된다. 이에 최후에 응고되는 부분을 조절하여 캐스트 내부에 결함이 생기지 않도록 하기 위하여 모울드의 형상을 변화시켜 보았다. 이는 사용자 인터페이스 프로그램의 첫 단계로 돌아가 다시 프로그램의 수행을 반복하는 것을 의미한다.

Fig.16은 캐스트의 윗부분을 단열시킨 변경된 형태의 모울드와 캐스트 형태에 대해 Fig.15의 해석에 적용되었던 바와 동일한 프로세스들을 반복 적용한 것이다. Fig.16에서는 변화된 모울드와 캐스트의 형태에 대한 해석 결과가 Fig.15에서 나타난 결과와는 다른 양상을 보여줌을 알수 있다. 유한요소 해석결과 시간에 따른 응고양상은 하단부의 플랜지부분부터 응고되는 것은 전과 동일하나 응고의 진행이 캐스트영역의 가운데 부분에서 끝나는 것이 아니라 단열처리된 캐스트의 최상부에서 응고가 끝남을 알 수 있다. 이는 캐스트영역의 중심부에서 일어나는 최종응고를 방지하고 이로 인하여 야기되는 결함의 발생가능성을 배제하고자 한 목적에 잘 부합하는 결과라 할 수 있다.

Fig.17은 모울드와 캐스트의 형상변화의 다른 한 예로 원하는 응고양상

을 얻기 위하여 모울드내에 냉각장치를 한 경우에 대해 고려하였다. 모울드 내에 25°C의 물을 냉매로 사용하는 냉각장치가 해석단면에 수직으로 무한한 길이로 놓여 있다고 가정하고, 냉매인 물의 온도는 일정하다고 보았다. 또한 모울드와 접하는 캐스트 상부는 단열이 되어 있다고 가정하였다. 그 해석 결과는 캐스트 하부에 위치한 냉각장치의 역할로 캐스트 하부가 빨리 냉각됨을 볼 수 있고, 또한 단열처리된 캐스트 상부로의 열유출이 적은 이유로 시간 경과후 응고양상은 캐스트상부에서 최후의 응고가 일어남을 알 수 있다.

이상에서 본 바와 같이 본 연구는 원하는 품질의 우수한 캐스팅제품을 생산하기 위하여 모울드와 캐스트의 형상변화, 경계조건변화 그리고 물성치의 변화 등으로 가장 적합한 캐스팅 조건의 도출을 사용자 인터페이스를 이용하여 구현하므로써 실제 현장의 근로자들이 신속하고 쉽게 사용할 수 있는 방향을 제시하였다.

본 해석에서는 편의상 라이저(Riser) 및 주입구(Runner) 등의 위치는 고려치 않았으나, 이와 같은 온도분포 해석을 이용하여 실제 주조공정시 용탕내부에서 최후에 응고되는 부분의 위치를 예측할 수 있으며, 따라서 라이저의 위치등을 적절히 조절하여 최후에 응고되는 부위에 기공이 발생하지 않도록 함으로써 결함이 없는 주조품 생산에 기여할 것으로 생각되며, 이외에도 국부적인 급속냉각이나 주형의 냉각효과등이 주조품 응고에 미치는 영향등도 수치해석을 이용하여 예측하는 것이 가능할 것으로 여겨진다.

Ⅳ. 결 론

주조공정시 금형의 온도분포 및 용탕의 응고과정 해석을 위한 사용자 인터페이스 프로그램 및 유한요소 프로그램 개발을 통해 금형에 내부에서의 용탕의 응고현상을 알 수 있었으며 이 결과를 통해 용탕이 최후에 응고되는 부분을 예측함으로써 주형의 설계시 금형 형상 선정 및 라이저 등의 위치선정에 유용하게 사용될것으로 생각된다.

본 연구는 또한 CAD 프로그램과 X 윈도우 시스템 프로그램기법을 이용하므로써 데이터 준비에 시간소모가 많은 수치해석 프로그램을 신속하고 편리하게 수행하도록 하고 해석결과 판단의 용이성도 함께 부여하였다. 본 연구에서와 같은 사용자 인터페이스를 이용한 신속한 주조공정해석과 그 결과 판정을 실제 주조현장에 적용함으로서 주조품의 품질 및 생산성 향상에 에 기여할 수 있을 것으로 생각된다.

V. 참고문헌

1) V. Paschki, "Studies on solidification of castings", Trans. of A.F.S., Vol 53, pp.90-101

2) A.B.Crowley, "Numerical solution of stefan problems", Int.J.heatmass transfer, vol.21, pp.215-219

3) C.Bonacina and G.Comini, "Numerical solution of phase-change problems", Int.J. Heat Mass Transfer, vol.16., pp.1825-1832

4) G. Comini and S. Del Guidice, "Finite element solution of non-linear heat conduction problems with special reference to phase change", Int. J. Num. Method in Eng., Vol.8, 1974, pp.613-624

5) K. Morgan, R.W. Lewis and O.C. Zienkiewicz, "An improved algorithm for heat conduction problems with phase change", Int.J.Num.Methods Eng., Vol.12, 1978,pp.1191 - 1195

6) W. Donald Rolph Ⅲ, "An efficient algorithm for analysis of nonlinear heat transfer with phase changes", Int.J.num.Methods eng., vol.18, 1982, pp.119-134

7) C.P.Hong, T.Umeda and Y.Kimura, "Numerical modeling for casting solidification : Part I. The coupling of the boundary element and finite difference methods for solidification problems", Metall.Trans.B, Vol.15B, No.3, 1984, pp.91-100

8) C.P.Hong,T.Umeda and Y.Kimura, "Solidification simulation of

shaped castings by the boundary element method and prediction of shrinkage cavity", 鑄物, Vol.59, No.2, 1987, pp.61-68, pp.91-107

9) 이영철, 이학주, "알루미늄합금주물의 응고해석과 수축공 예측", 대한금속학회지, Vol. 26, No.8, 1988, pp.770-775

10) A.J.Dalhuijsen and A.Segal, "Comparison of finite element techniques for solidification problems", Int.J.Num.Method in Eng., Vol.23, 1986, pp.1807-1829

11) 이 진형, 허훈, "FDM 및 FEM에 의한 금형주물 응고의 해석", '87 과기대 연구보고서 K-87-00052, 1987

12) J. Yoo and B. Rubinsky, "A Finite-element method for the study of solidification processes in the presence of natural convection", Int.J.Num.Method in Eng., Vol.23, 1986, pp.1785-1805

13) K. Tamma and R. Namburu, "Prediction of thermal stress and deformation due to phase change in solidifying objects via flux/stress based finite element representations", Int.J.Num.Method in Eng., Vol 29, 1990, pp.1473-1485

14) R.Song, G.Dhatt and A.Cheikh, "Thermo-mechanical finite element model of casting systems", Int.J.Num.Method in Eng., Vol.30, 1990, pp.579-599

15) G.Dhatt, D.M.Gao and A.Cheikh, "A finite element simulation of metal flow in moulds", Int.J.Num.Method in Eng., Vol.30, 1990, pp.821-831

16) N.Barkakati, X-Window System Programming , SAMS, 1990

17) N. Kikuchi, Finite element methods in mechanics , Cambridge Unv.

Press, 1986

18) T. Tanikuchi, "An interactive automatic mesh generator for the micro computer" , Computers & Structures, vol.30, No.3, 1988, pp.715-722

19) J.P. Holman, Heat Transfer , 5th. ed. pp.249-276

20) Y. Nishida, W. Droste, "The air-gap formation process at the Casting-Mold interface and the heat transfer mechanism in through the gap", Metallurgical Transactions B, Vol.17B, 1986, pp.833-844

21) L. Fletcher, "Recent Developments in contact Conductance heat transfer", Trans.of ASME Journel of heat transfer, vol.110, 1988, pp.1059-1070

22) E. Brandes, Smithells Metals Reference Book, 2nd ed. Butterworths, 1980, pp.8-1,41

23) D. Gaskell, Introduction to metallurgical thermodynamics, 2nd ed. McGraw-Hill, 1981, pp.540, 587-590

Table 1 : Comparison of properties of air and water

	ρ $\frac{kg}{m^3}$	C_p $\frac{J}{kg\ K}$	ν $\frac{m^2}{s}$	k $\frac{W}{m\ K}$	a $\frac{m^2}{s}$	Pr	β $\frac{1}{k}$
air(20°C)	1.177	1005	15.68	0.0262	22.1	0.708	36.7
water(20°C)	1000.5	4179	1.006	0.597	0.14	7.02	1.00

Table 2 : Values of Pr and F1(Pr)

Pr	0.01	0.72	0.733	1.0	2.0	10	100
F1(Pr)	0.0812	0.5046	0.5080	0.5671	0.7165	1.1694	2.191

Table 3 : h_c for several types of cast-mold boundary and flow pattern

boundary	laminar $10^4 < Ra < 10^9$	turbulent $10^9 < Ra$
vertical plate or cylinder	$1.42(\Delta T/L)^{1/4}$	$0.95(\Delta T)^{1/3}$
horizontal cylinder	$1.32(\Delta T/d)^{1/4}$	$1.25(\Delta T)^{1/3}$
horizontal plate	$1.32(\Delta T/L)^{1/4}$	$1.43(\Delta T)^{1/3}$
horizontal pplate	$0.62(\Delta T/L^2)^{1/4}$	
L: lenth of body, d: diameter(m)		

Under X Window System Environment

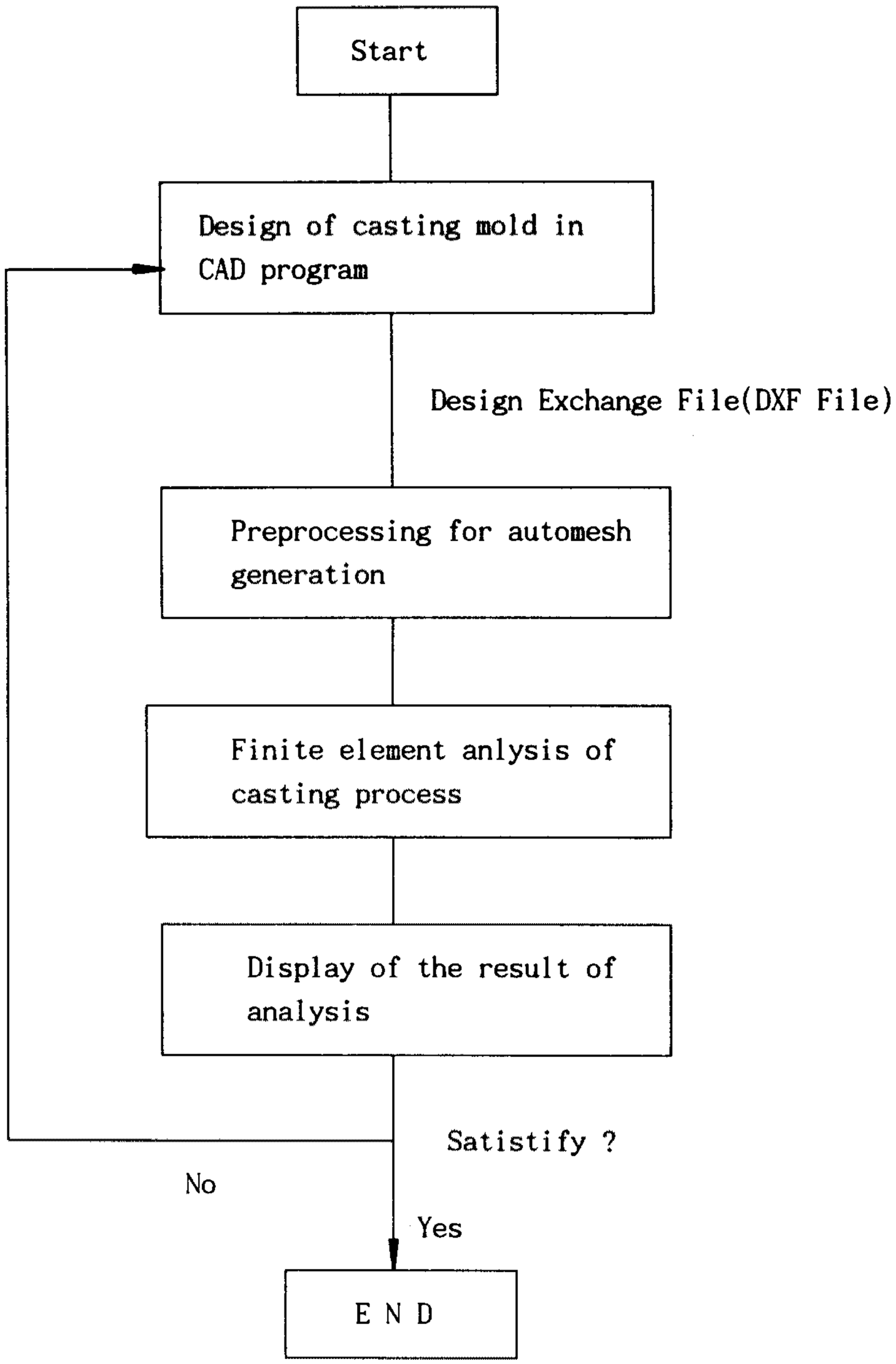


Fig. 1 Flow chart of the program for decision of optimal casting process variable

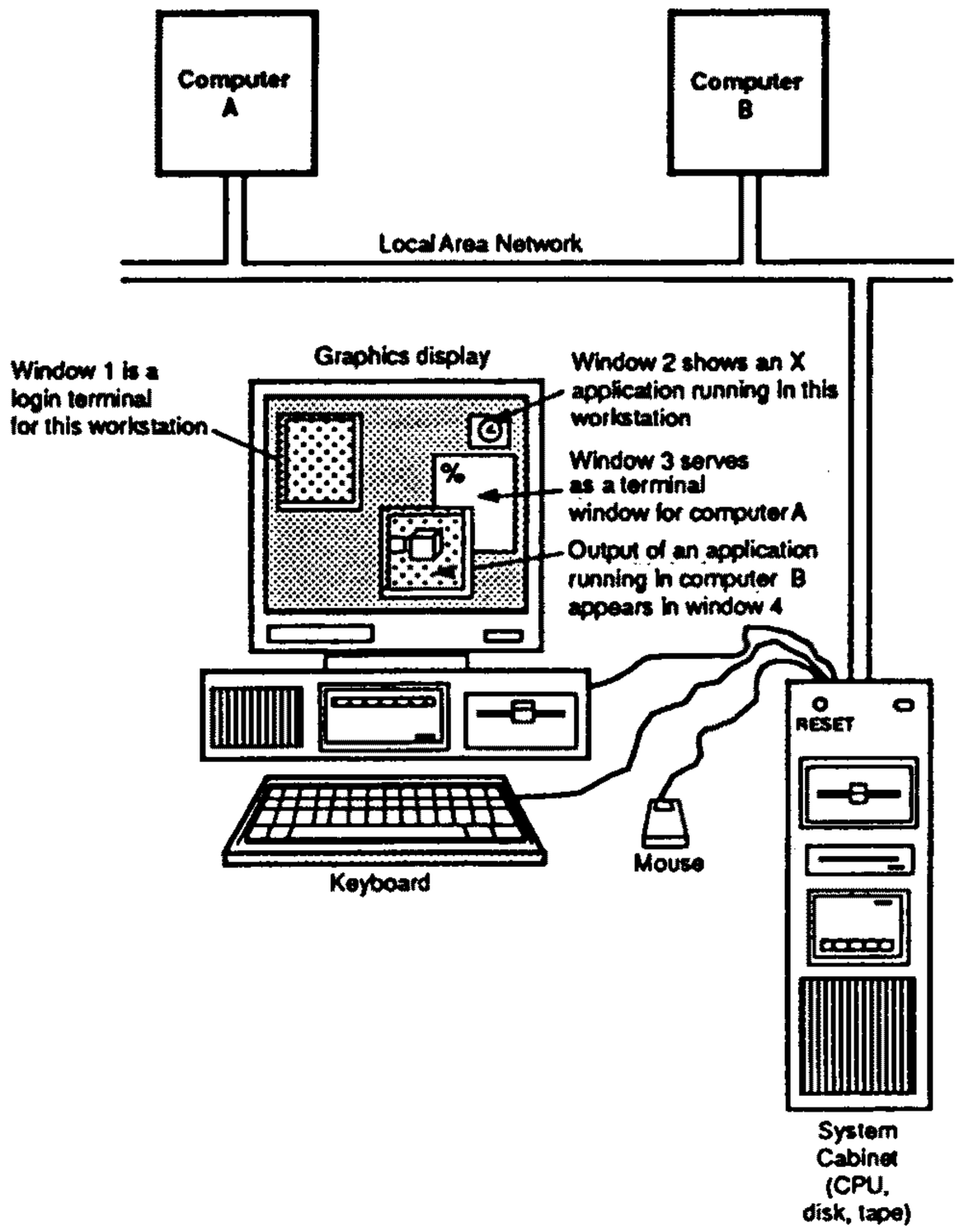


Fig.2 Workstation with X Window System

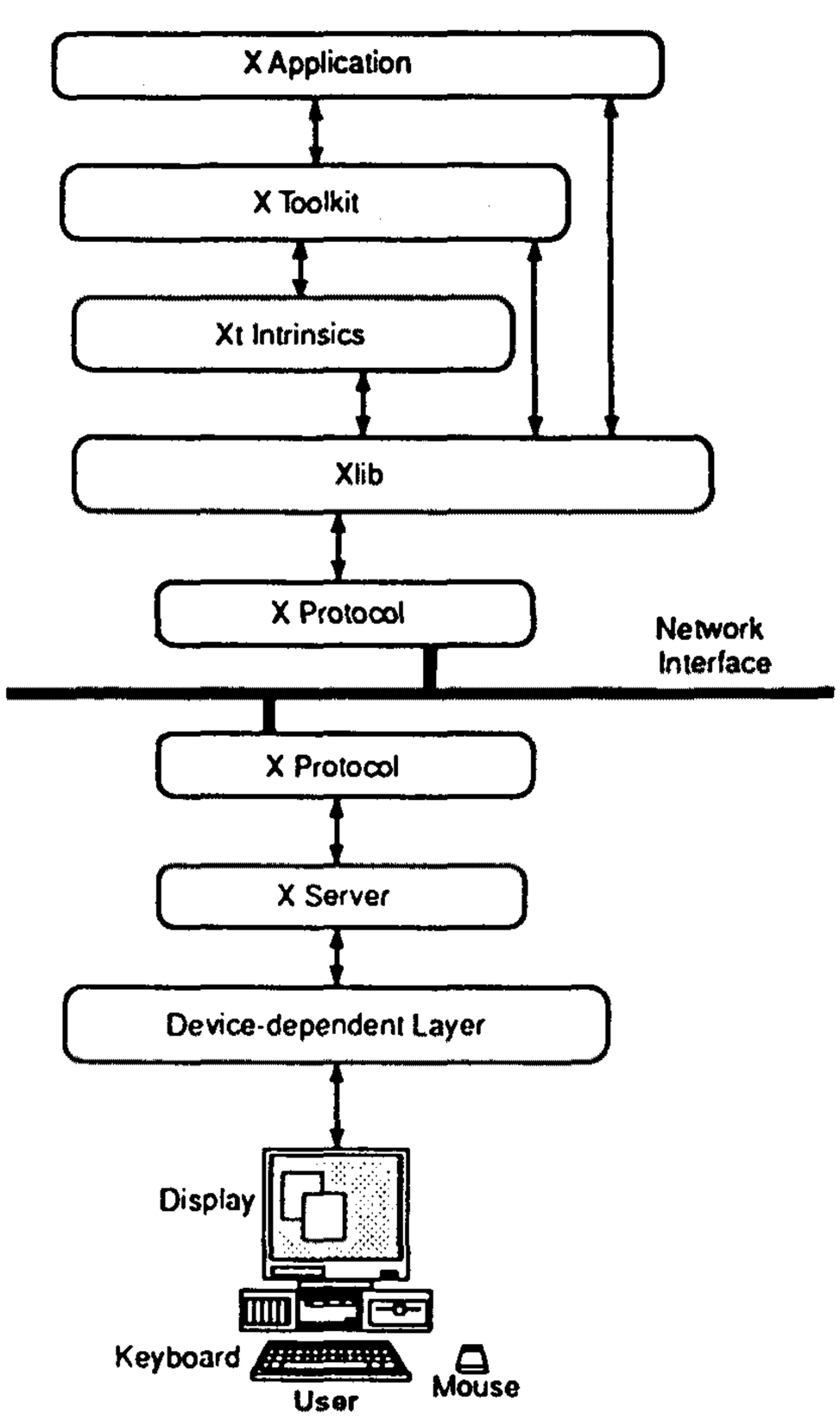


Fig.3 General structure of X application

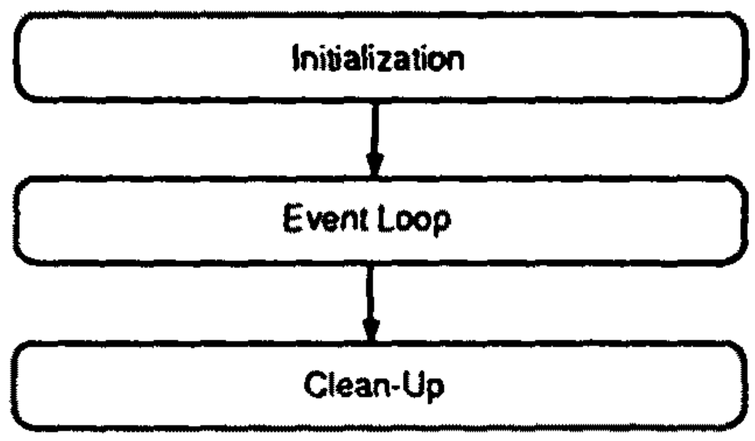


Fig.4 General structure of an event-driven program

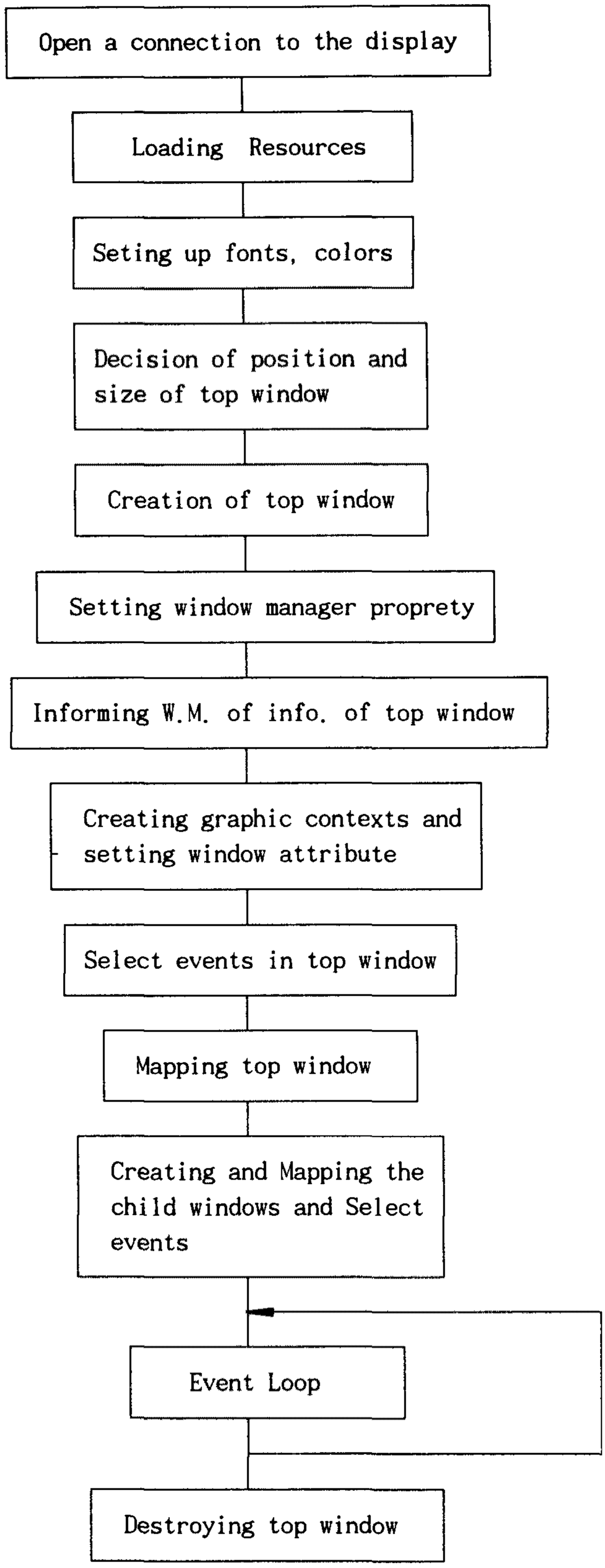
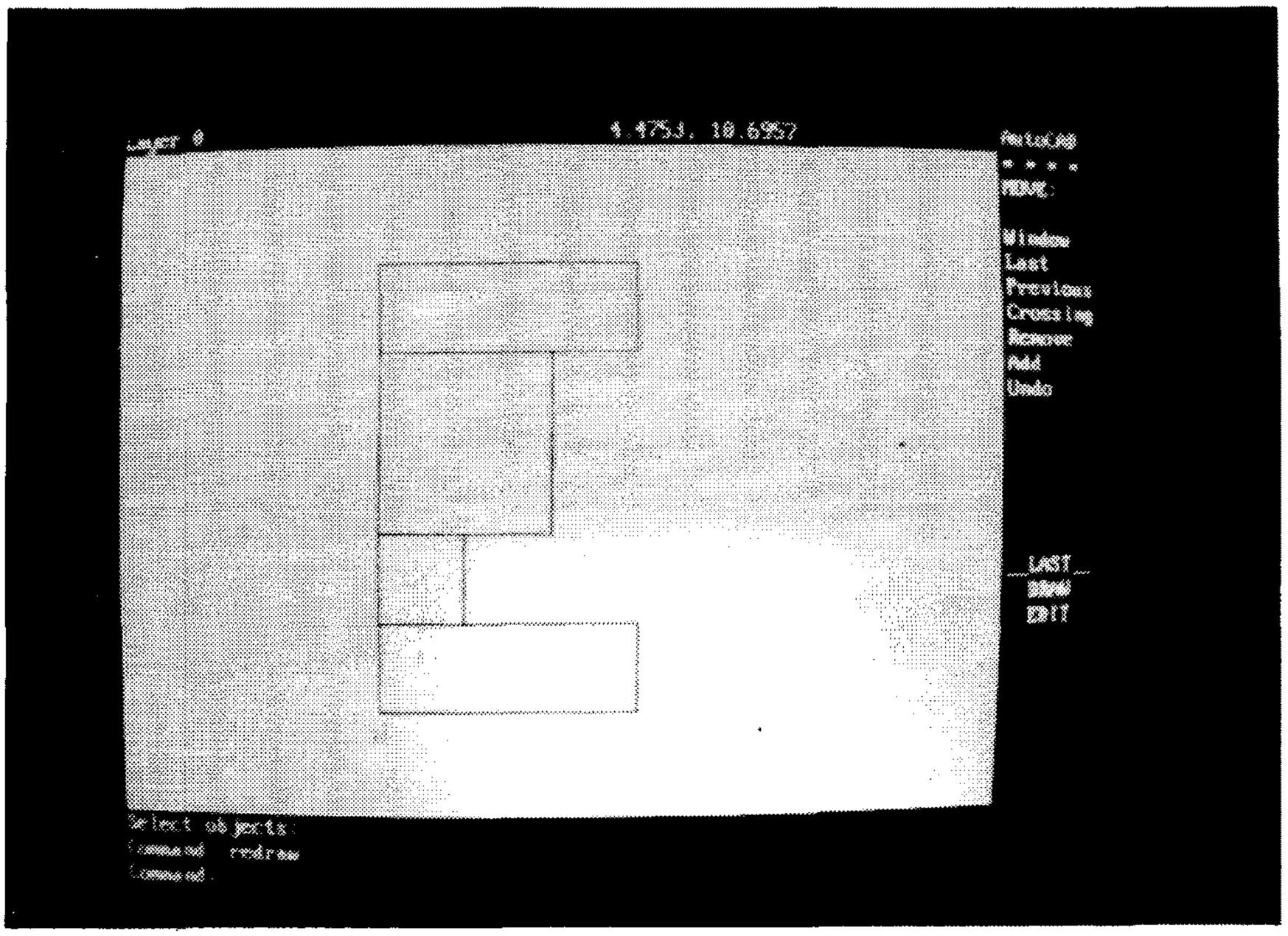


Fig 5. Flow chart of X Library programming



```
  0                                   3.0
SECTION                               31
  2                                   0.0
HEADER                                 0
   9                                  LINE
$ACADVER                               8
   1                                  0
AC1006                                 10
   9                                  9.0
$INSBASE                               20
 10                                   12.0
0.0                                    30
 20                                   0.0
0.0                                    11
 30                                   9.0
0.0                                    21
   9                                  15.0
$EXTMIN                                31
 10                                   0.0
0.0                                     0
 20                                   ENDSEC
0.0                                     0
   9                                  EOF
```

Fig.6 Drawing of CAD program and its DXF file

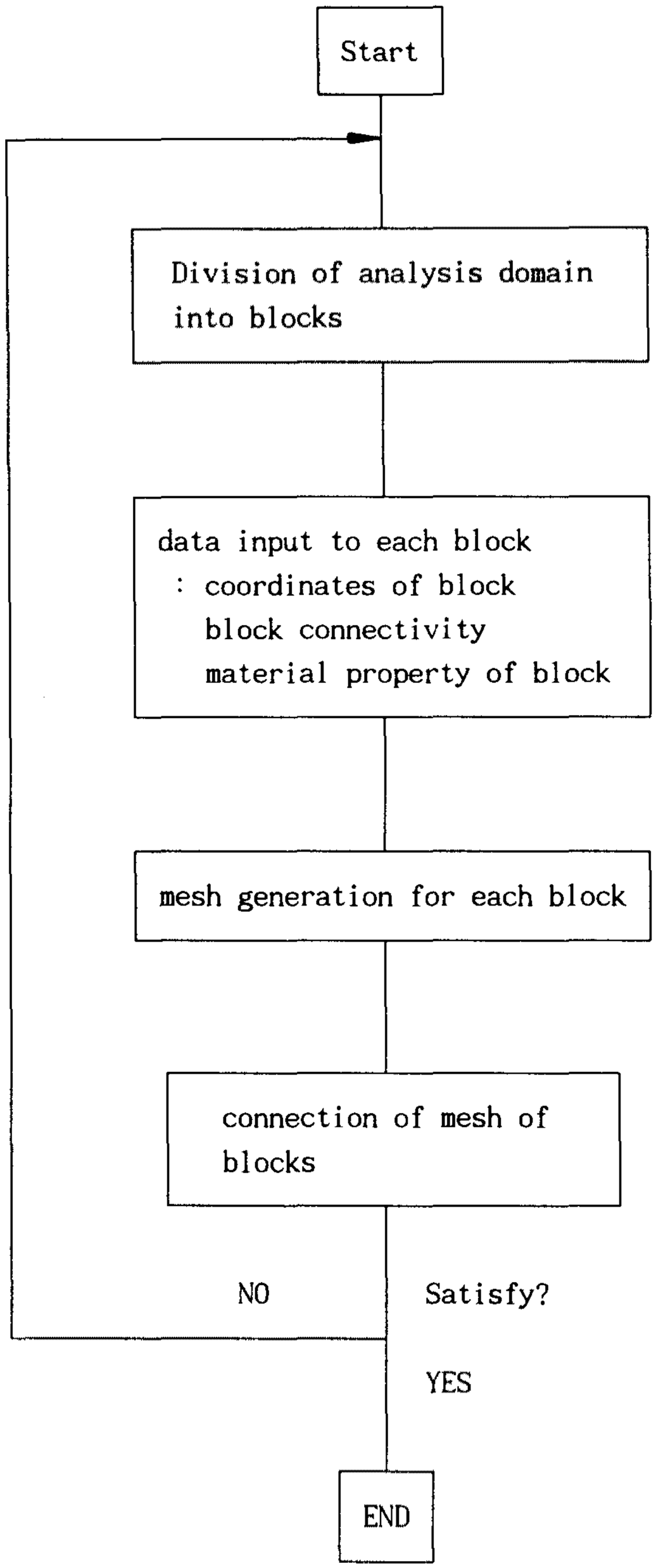
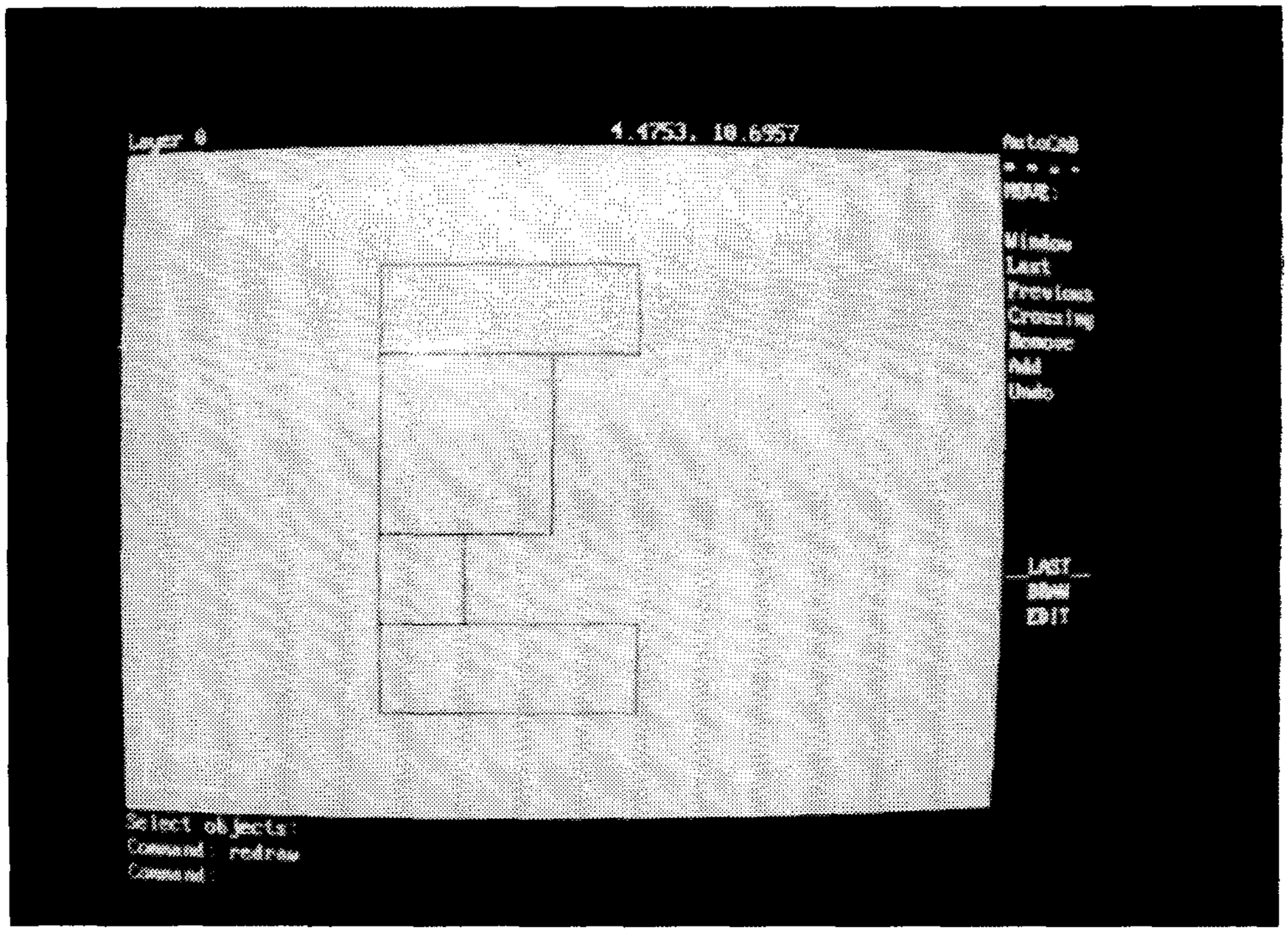
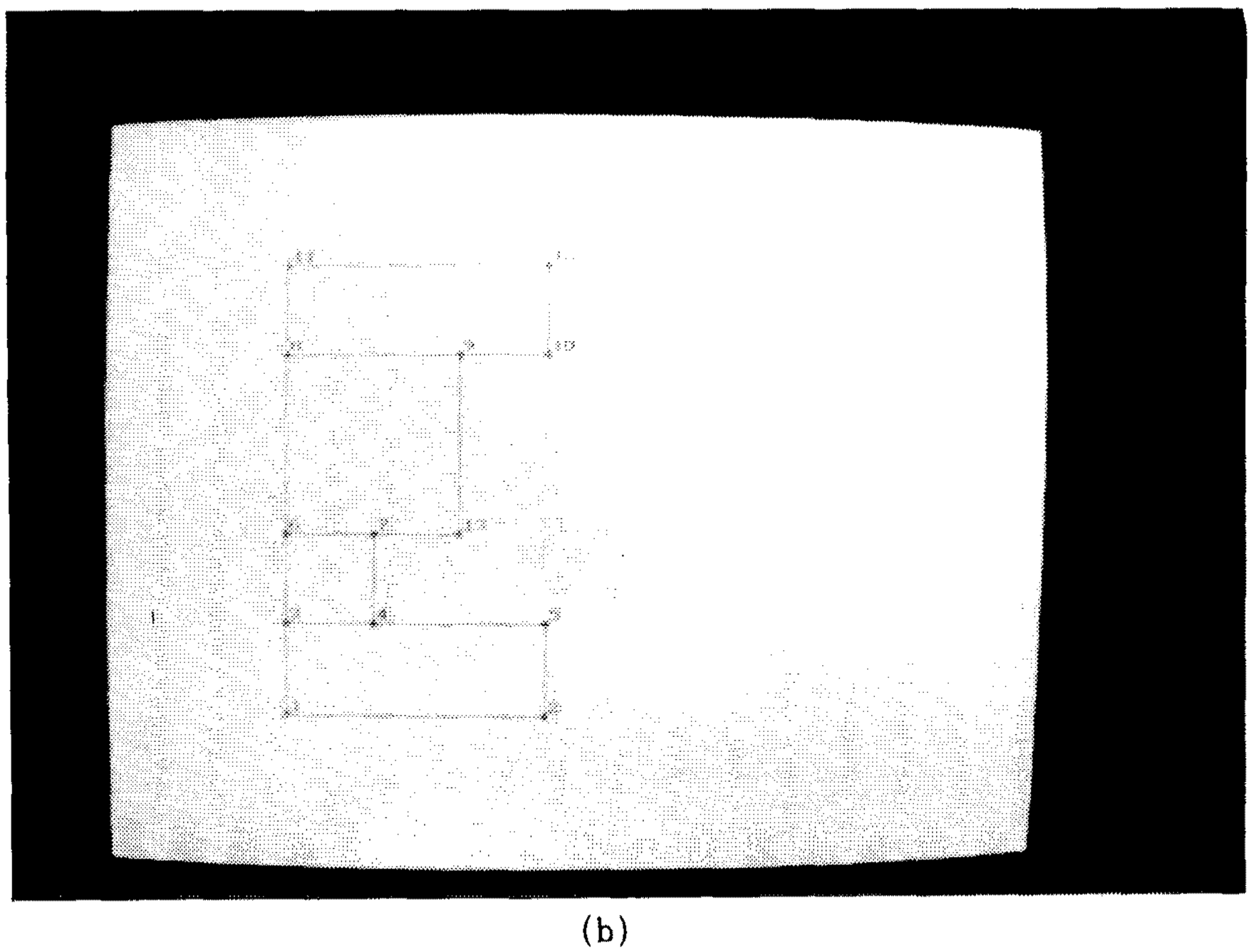


Fig.7 Flow chart of block-based algorithm

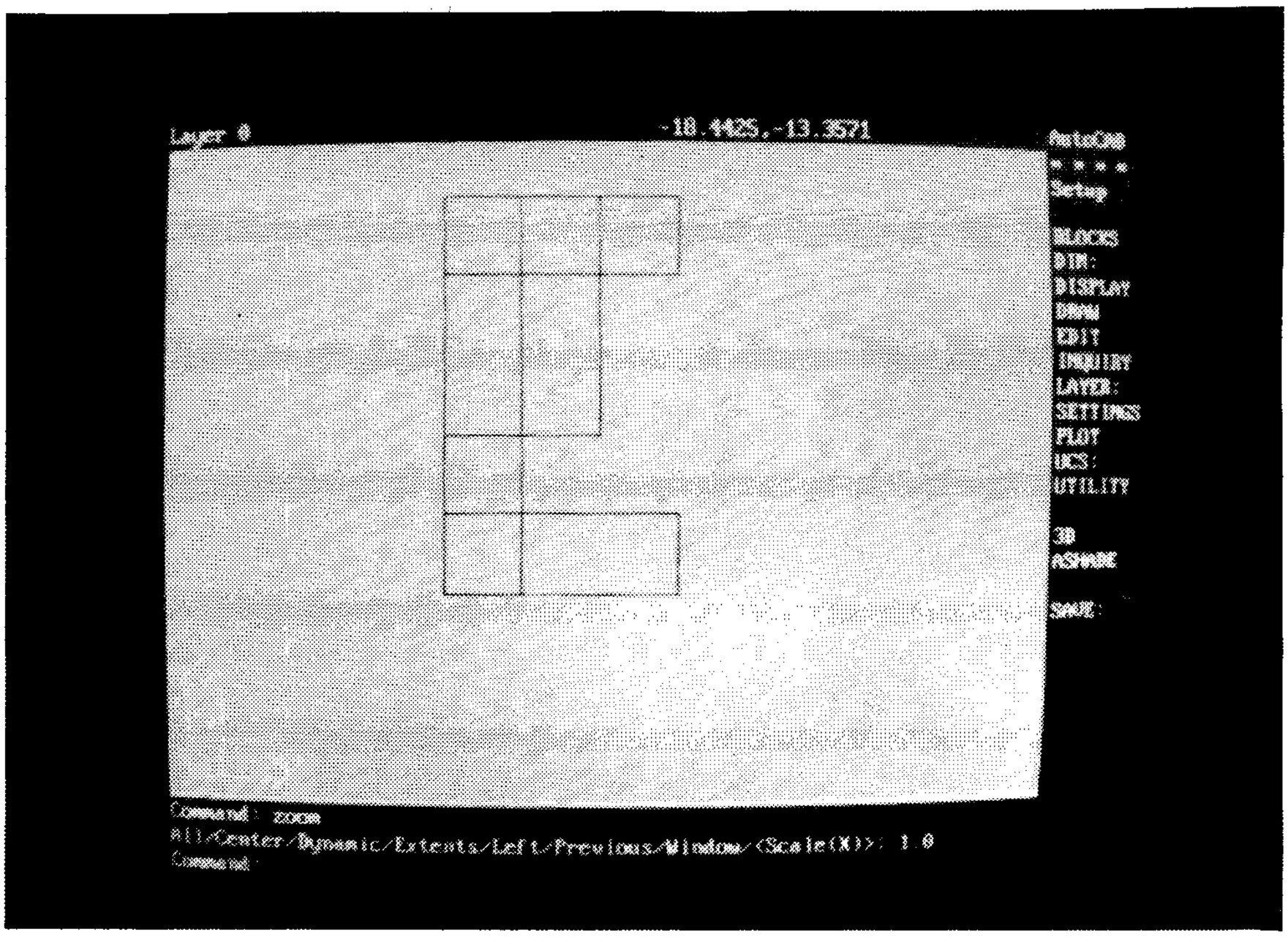


(a)

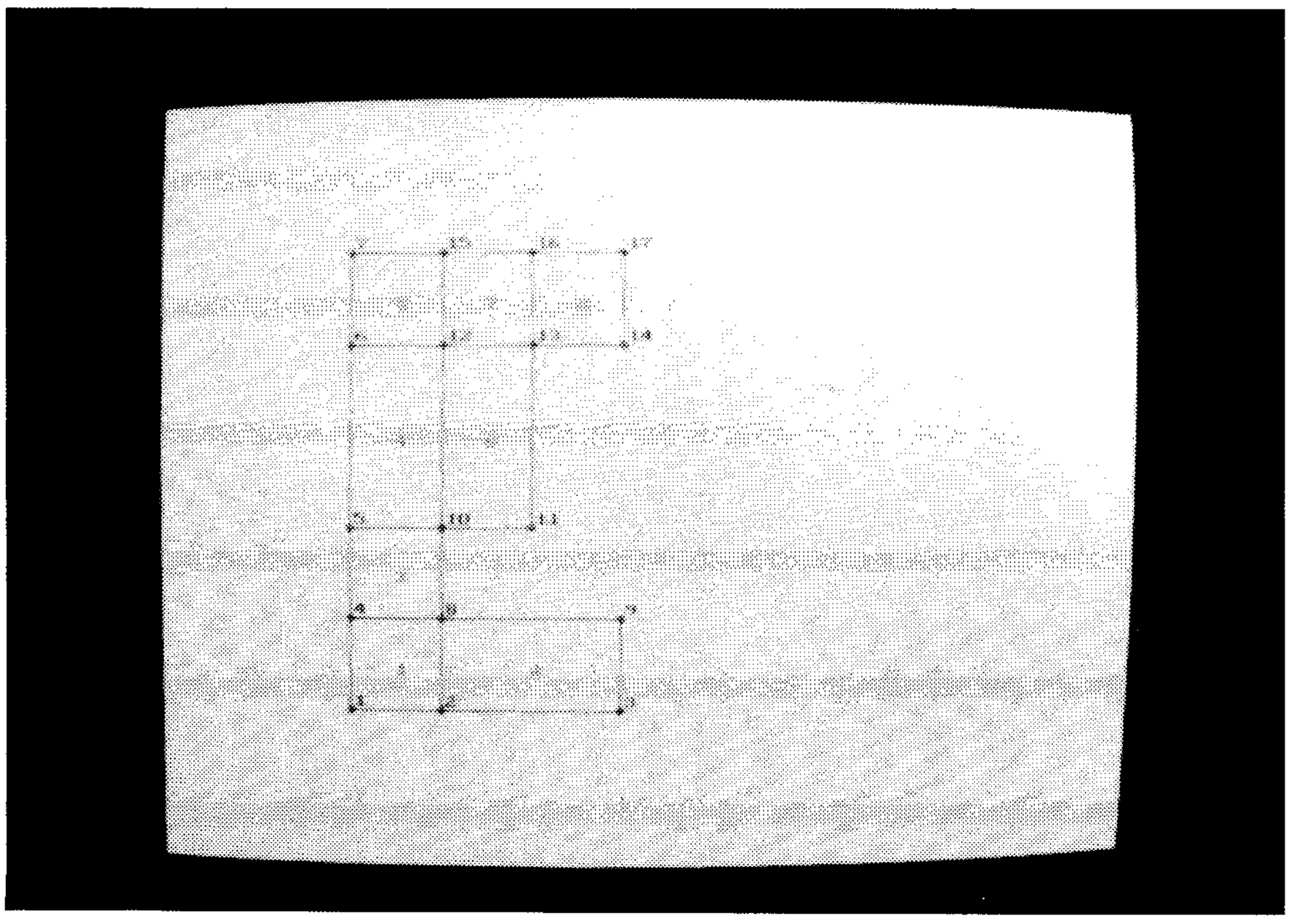


(b)

Fig.8 Results of preprocessing
(a)Drawing of CAD program
(b)Auto-block generation



(a)



(b)

Fig.9 Results of preprocessing
(a)Drawing of CAD program
(b)Auto-block generation

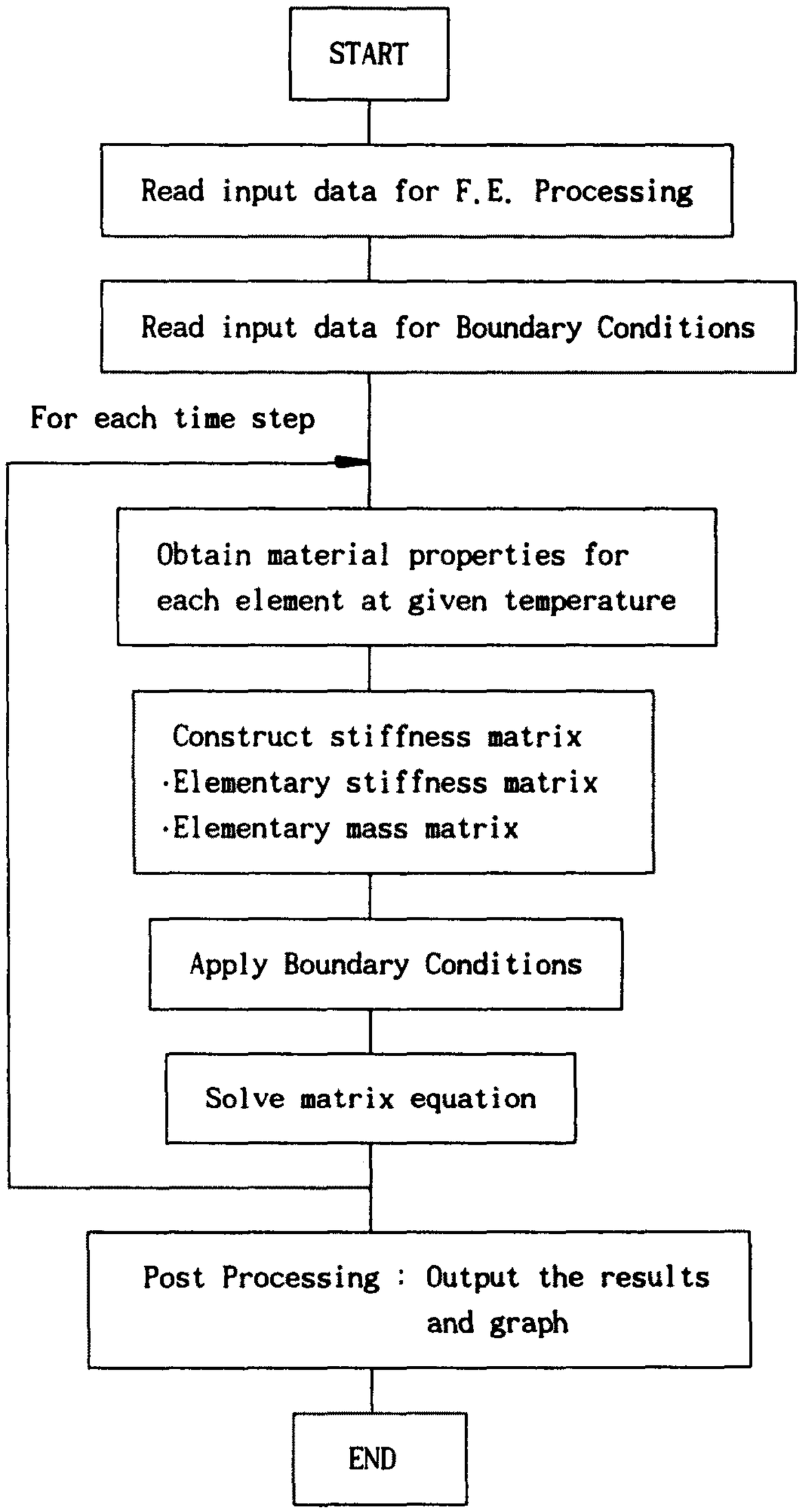
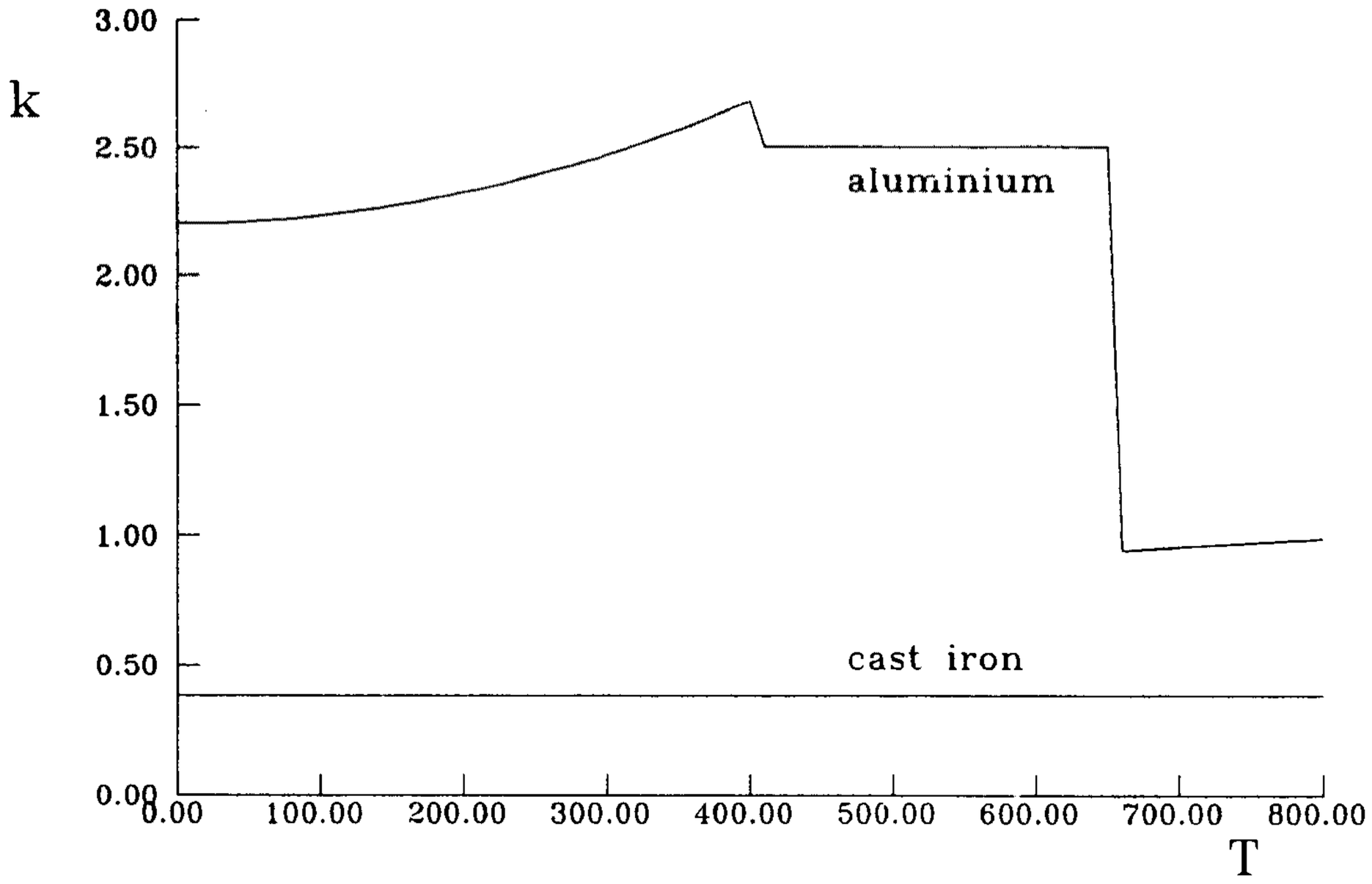
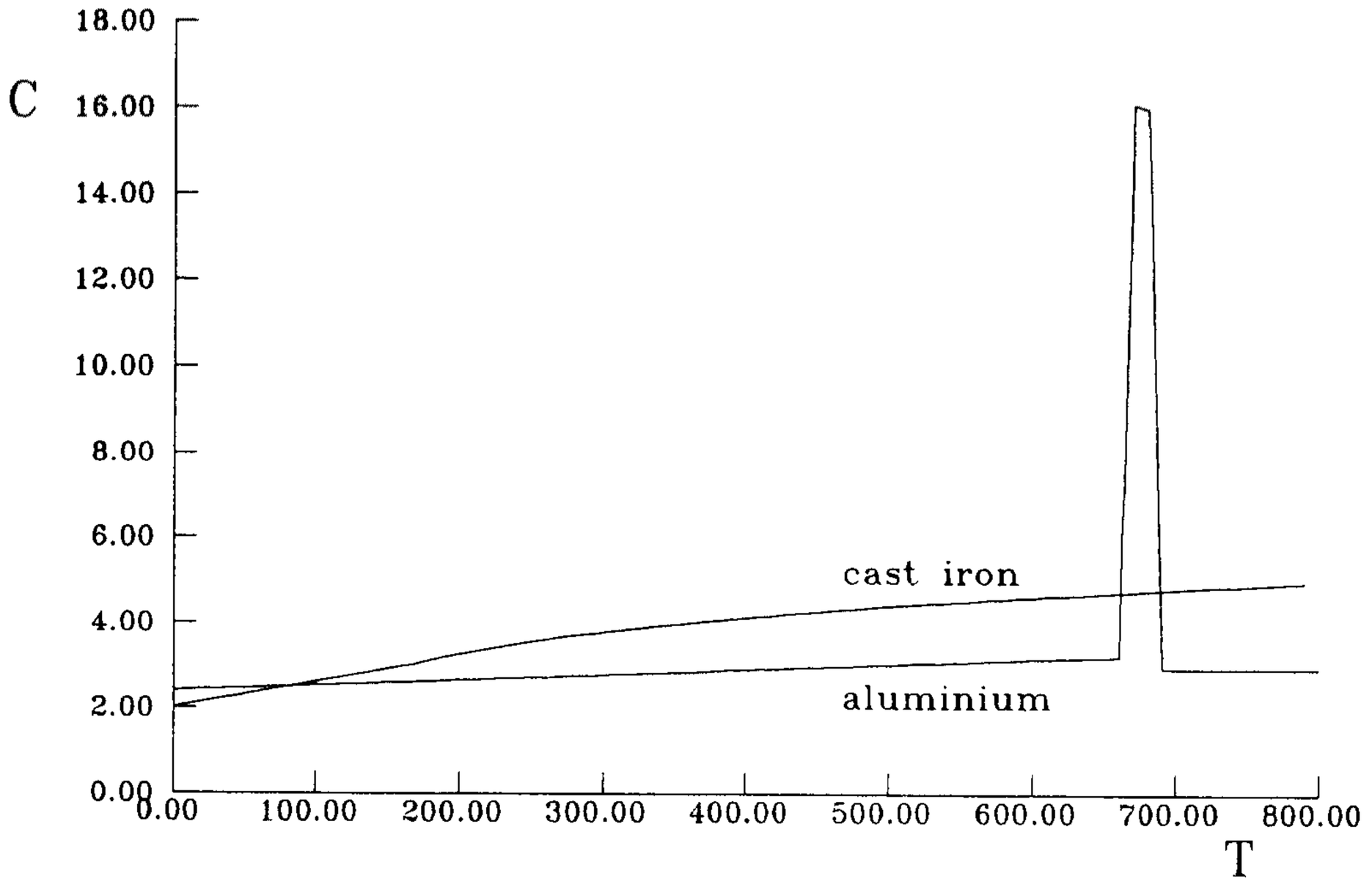


Fig.10 Flow chart of numerical analysis of casting process



(a)

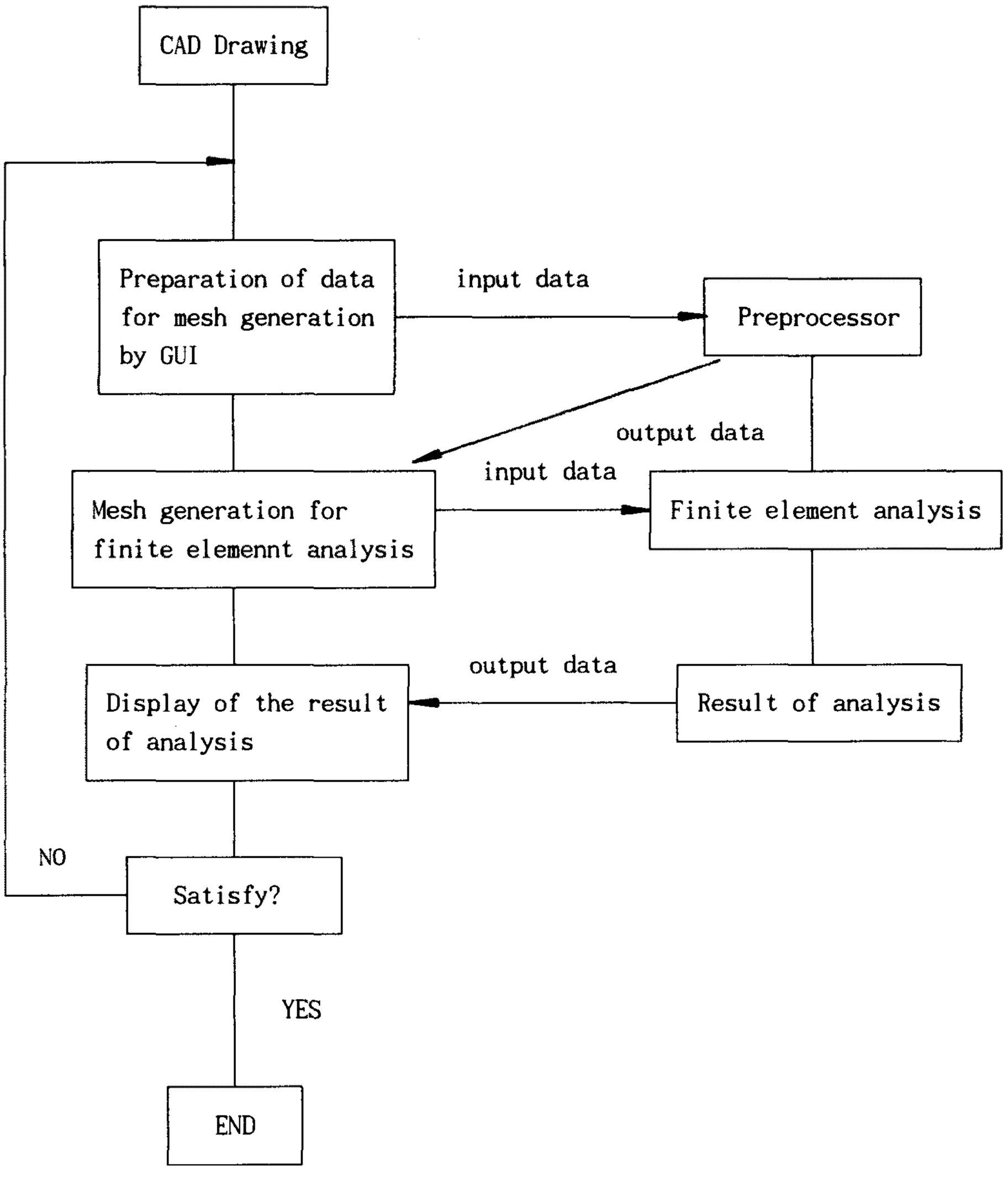


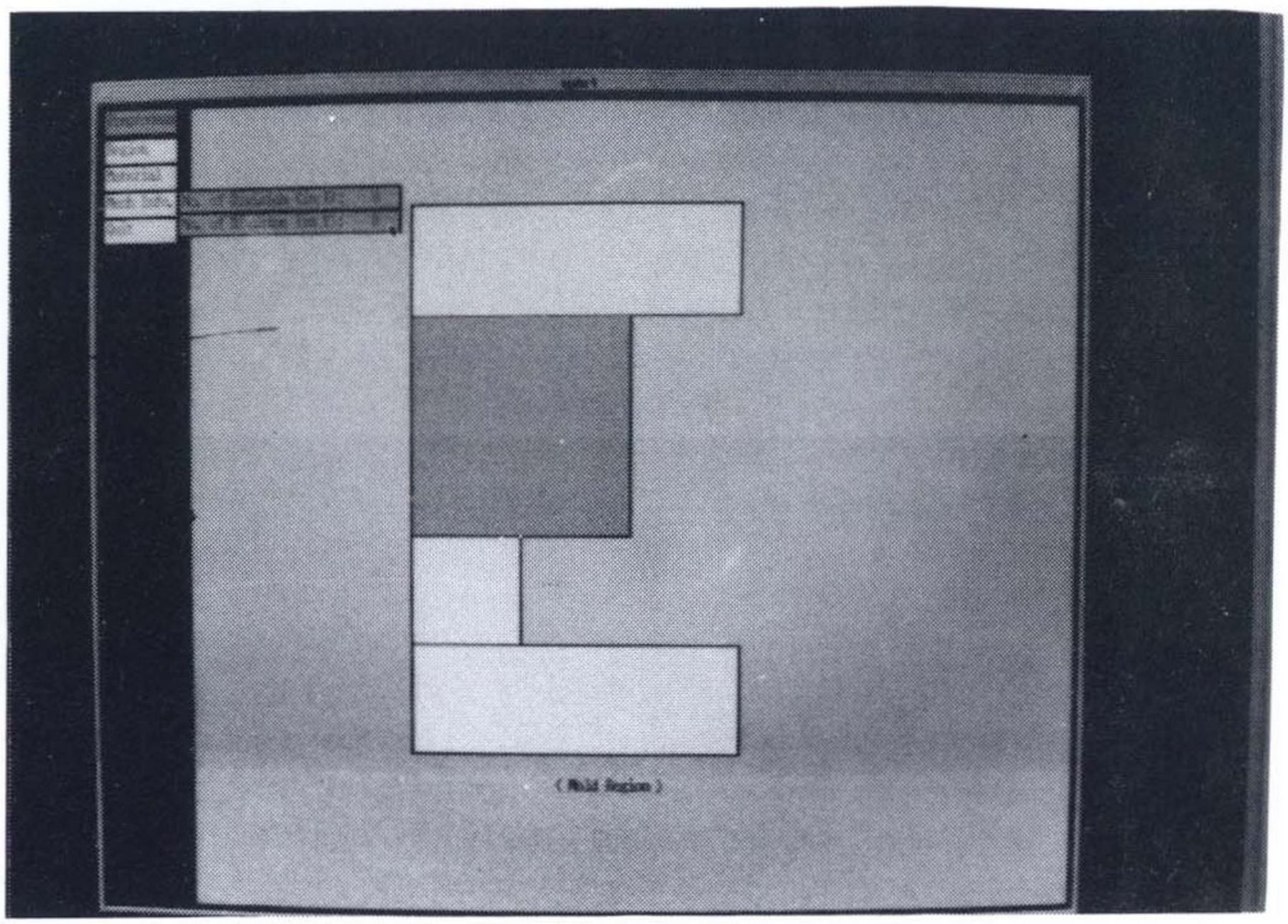
(b)

Fig.11 Material properties of castiron and aluminum
(a) Thermal conductivity
(b) Specific heat

X Library program

Analysis program





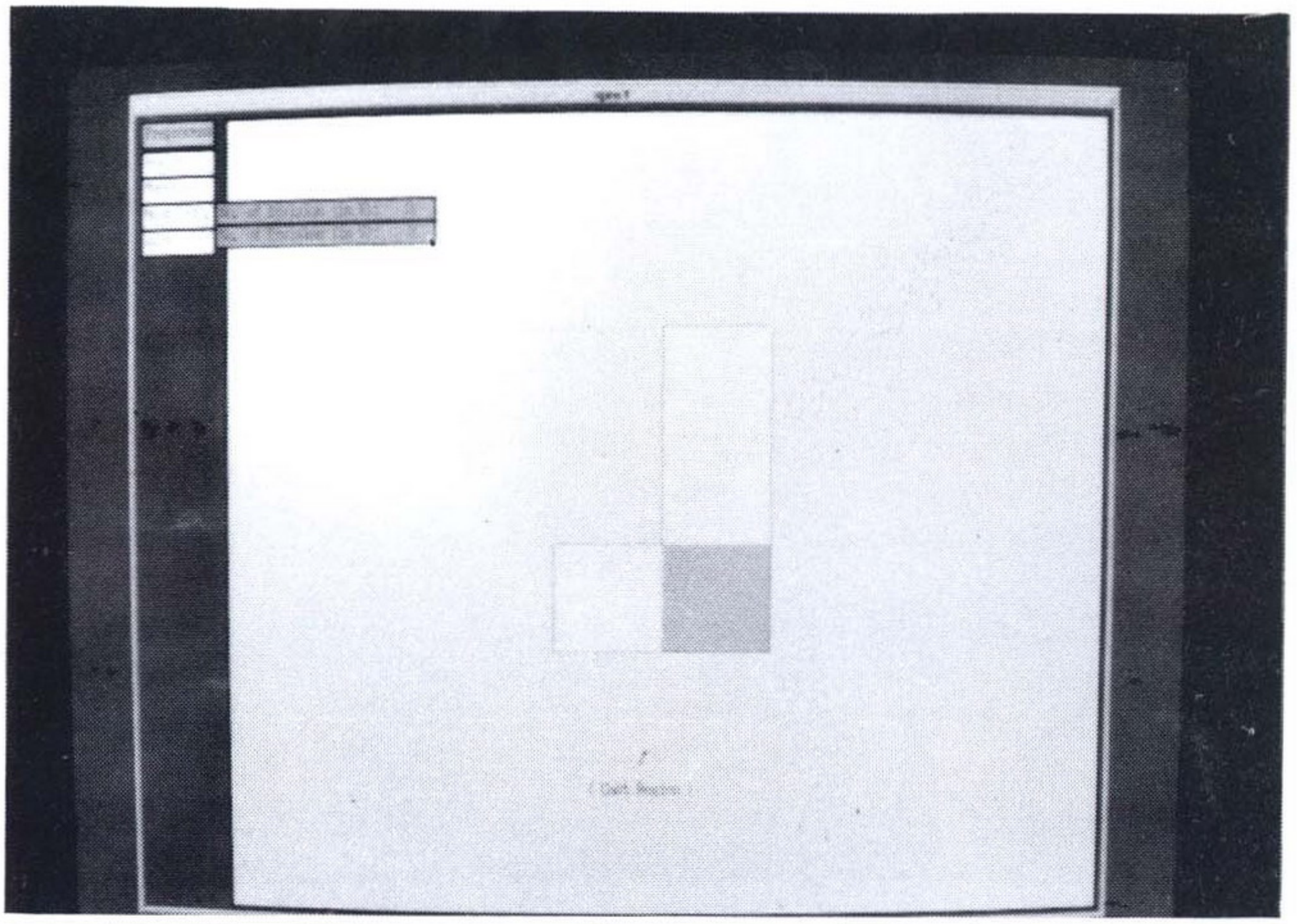


Fig.13 Process for mesh generation by GUI

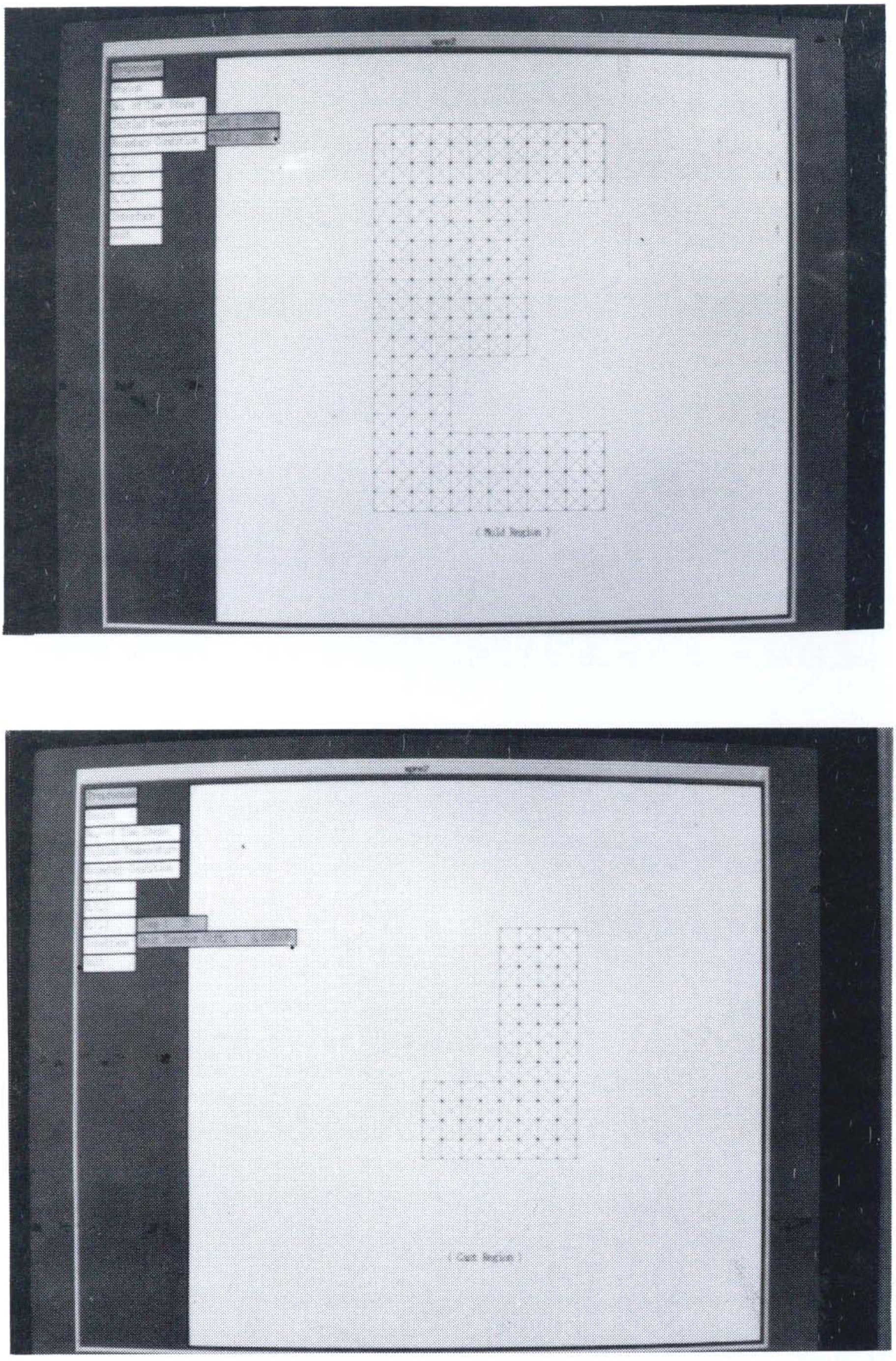
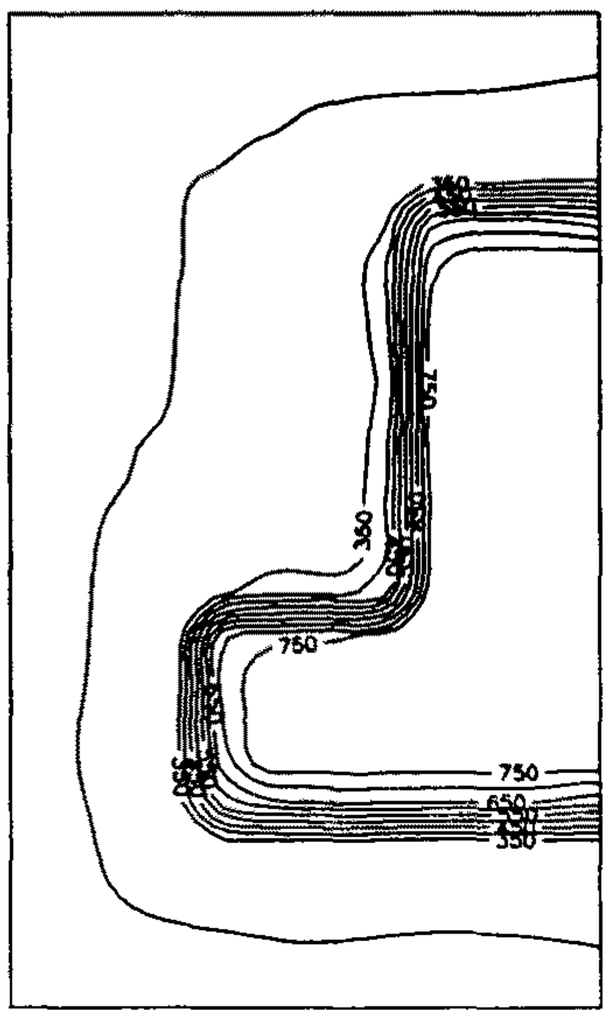
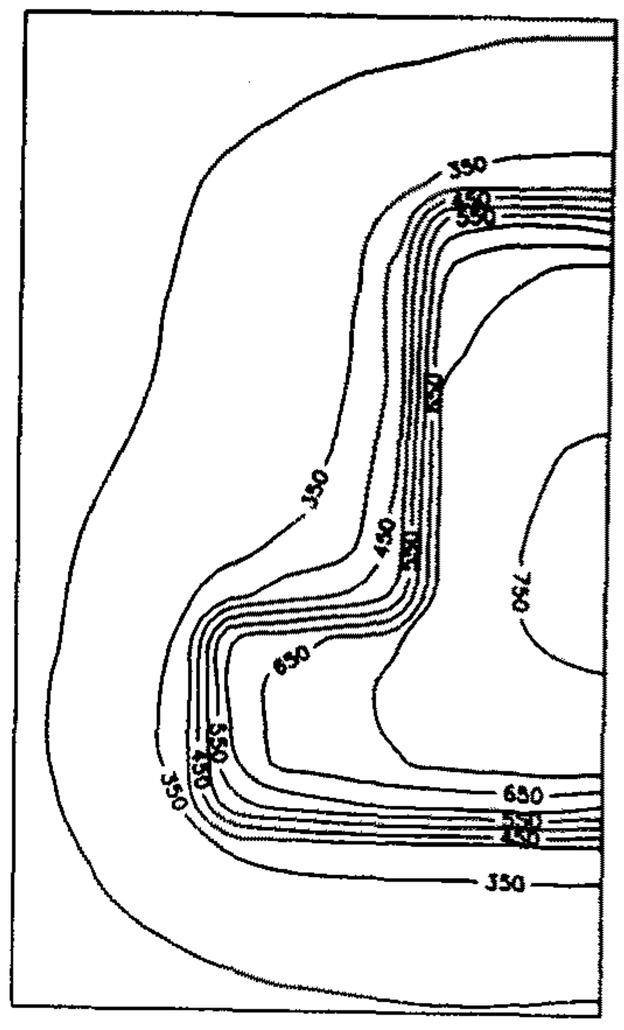


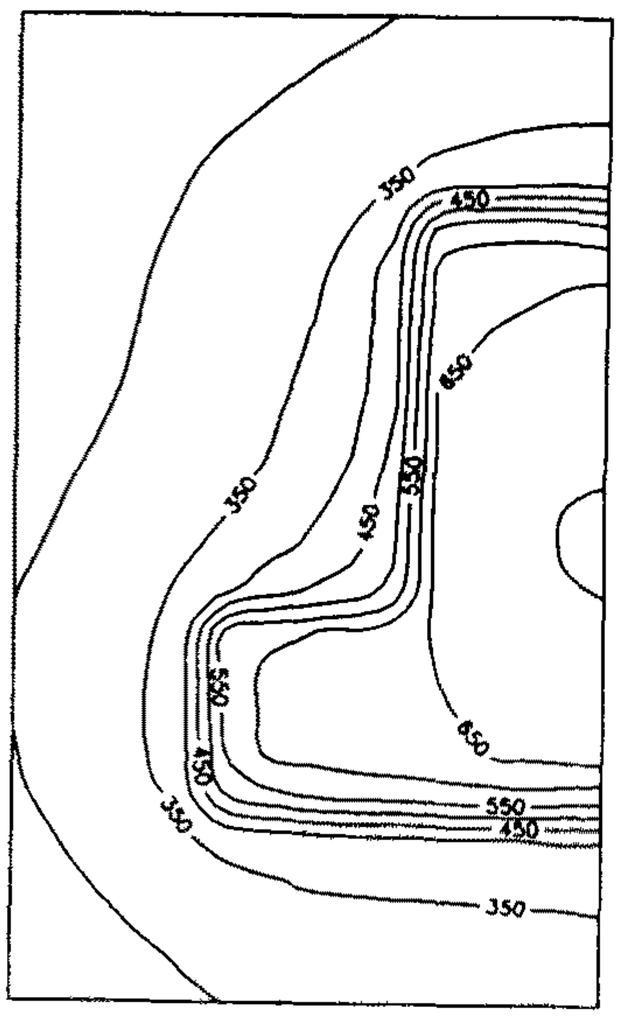
Fig.14 Completion of mesh generation and process for finite element analysis by GUI



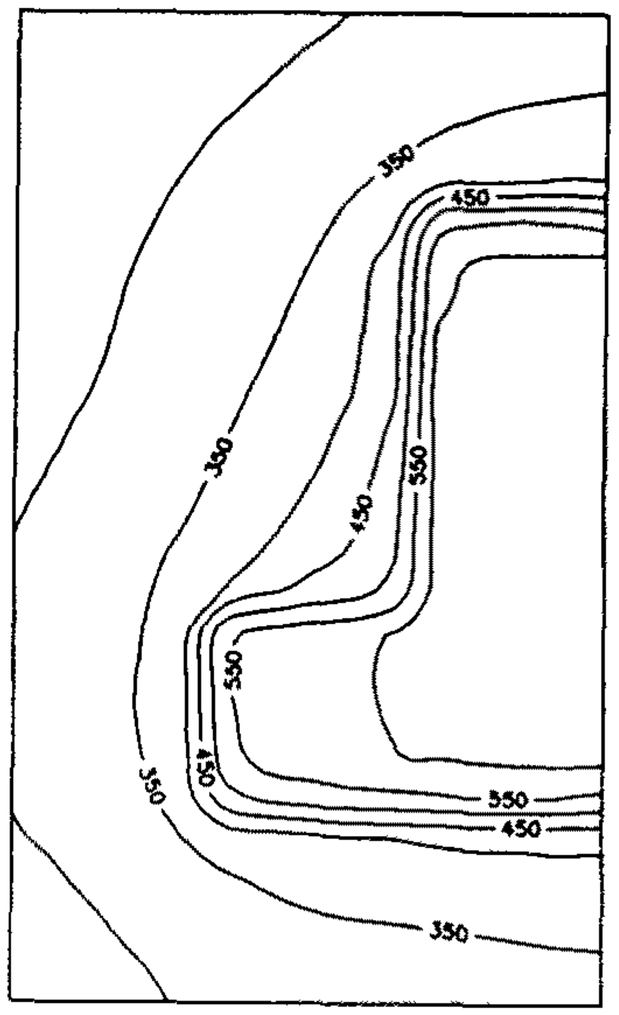
(a)



(b)

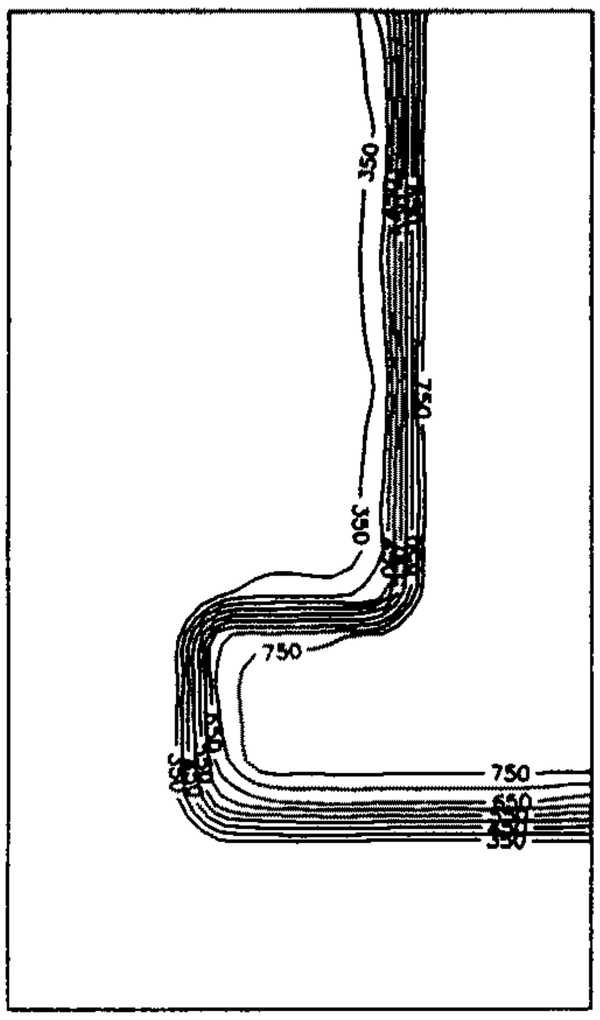


(c)

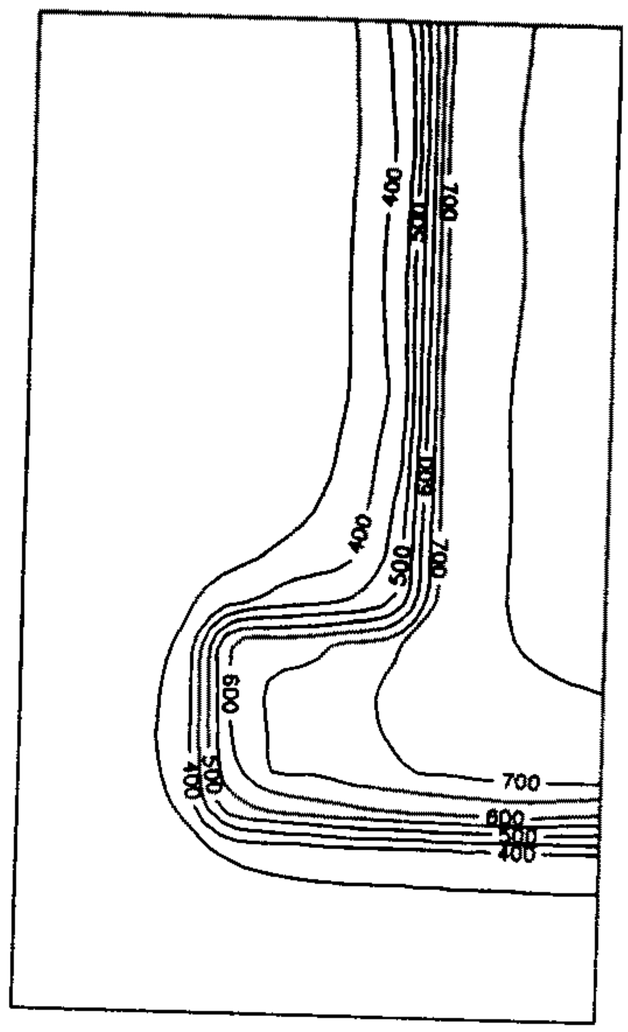


(d)

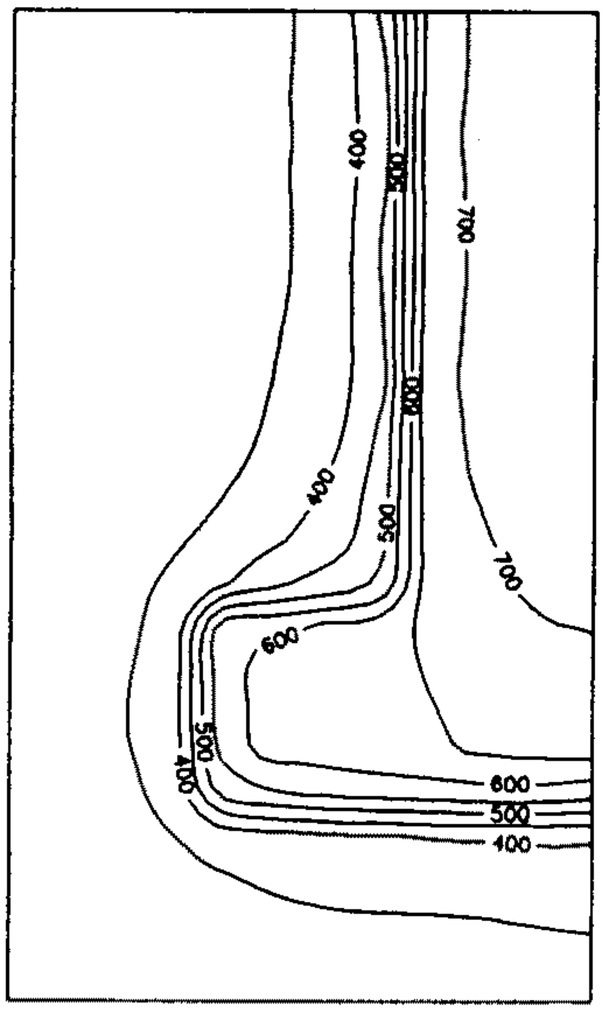
Fig. 15 Isothermal lines (a) t = 1 sec (b) t = 11 sec
(c) t= 21 sec (d) t = 31 sec



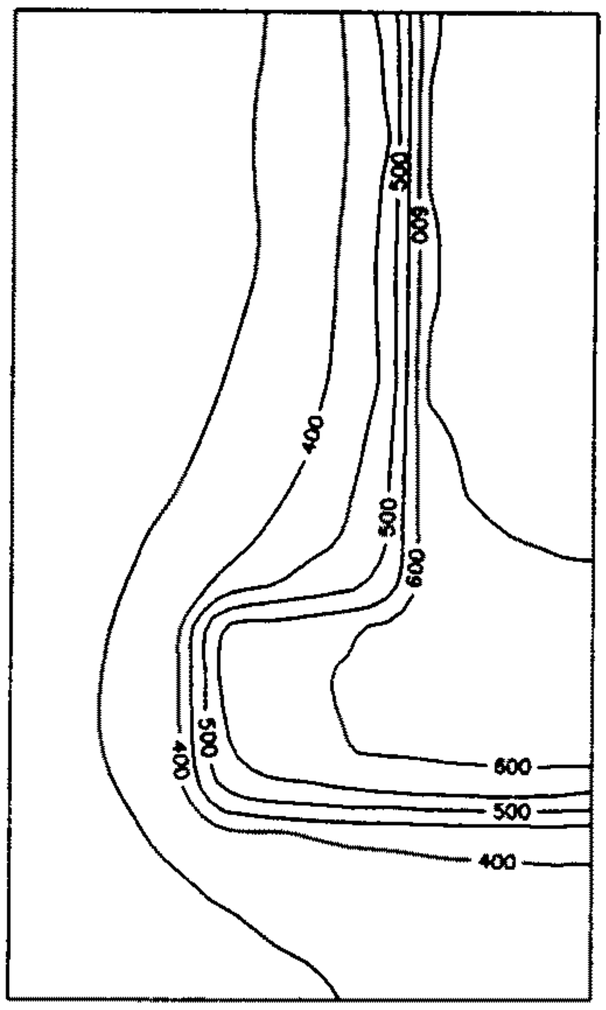
(a)



(b)

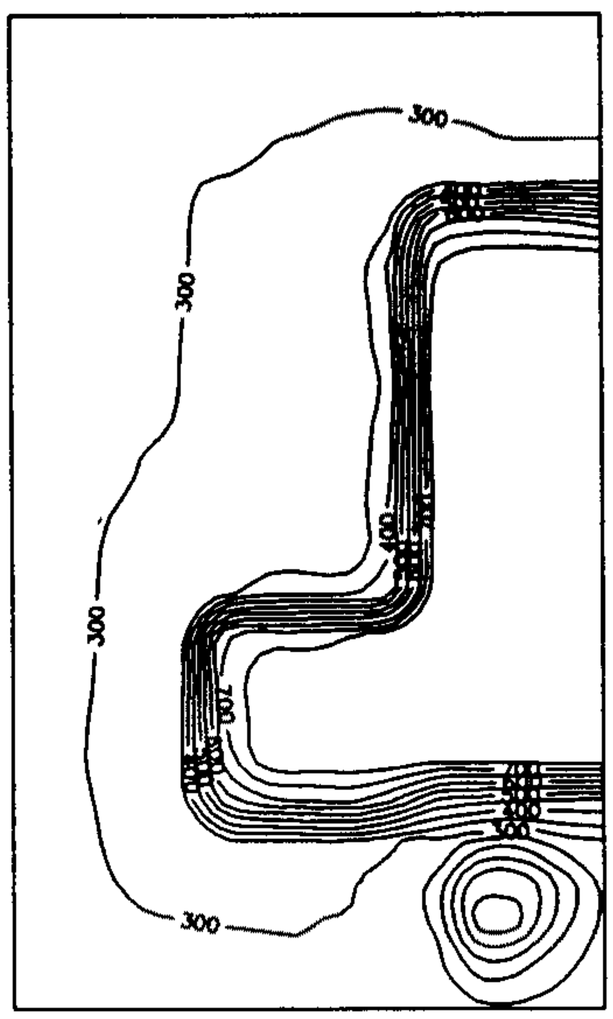


(c)

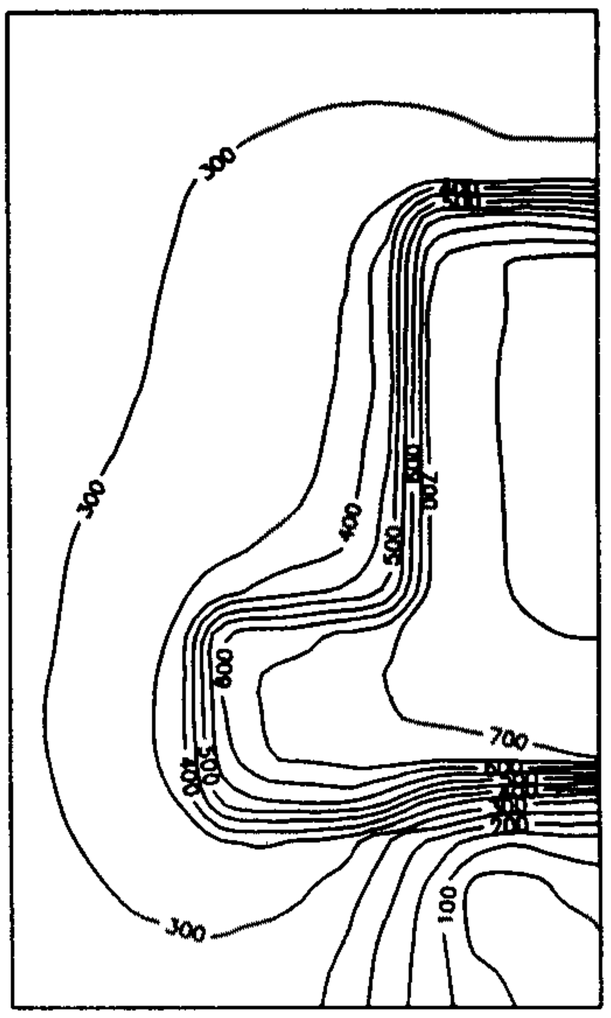


(d)

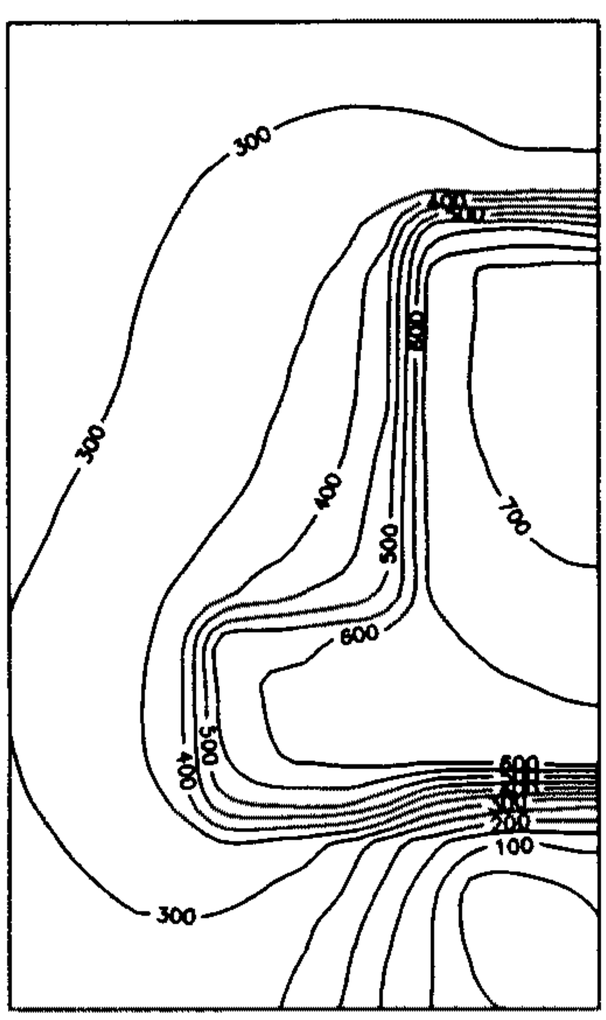
Fig. 16 Isothermal lines (a) t = 1 sec (b) t = 11 sec
(c) t= 21 sec (d) t = 31 sec



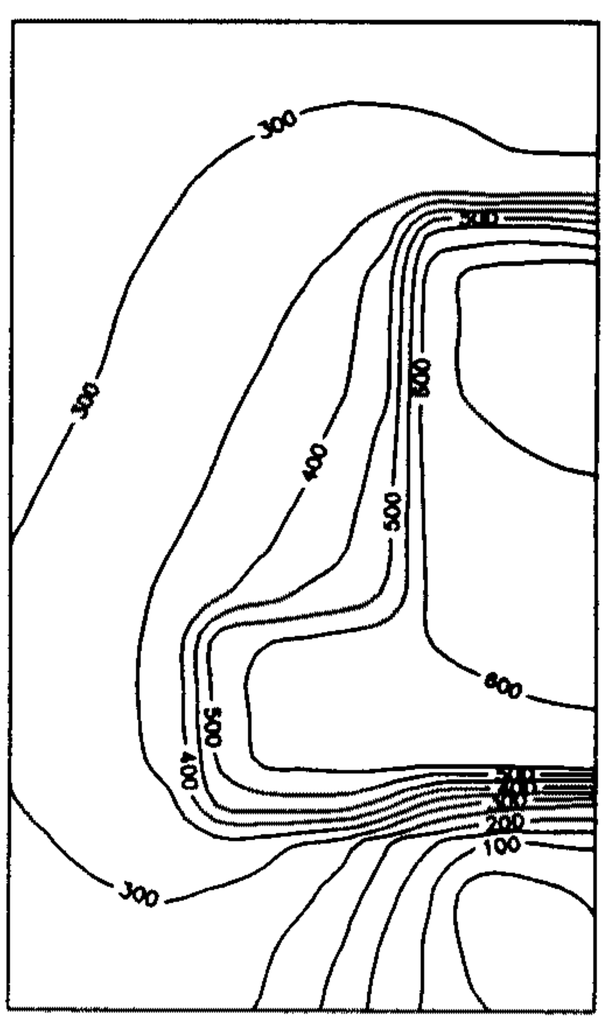
(a)



(b)



(c)



(d)

Fig. 17 Isothermal lines (a) t = 1 sec (b) t = 11 sec
(c) t= 21 sec (d) t = 31 sec

여 백

VII. 부록

부록 1 : CAD 프로그램에서 완성된 파일을 자동격자생성을 위한 화일로 변형하는 프로그램 (C language 사용)

여 백

```
#include <stdio.h>
#include <string.h>
#include <math.h>
#include <X11/Xlib.h>
#include <X11/Xutil.h>
#include <X11/keysym.h>
#define DEFAULT_MAIN_TEXT       "TEMPERATURE DISTRIBUTION"
#define DEFAULT_BGCOLOR         "light sky blue "
#define DEFAULT_FGCOLOR         "black"
#define DEFAULT_BDWIDTH         2
#define DEFAULT_FONT                "fixed"
#define NMAX 100

typedef struct XHELLO_PARAMS
{
    char *name;
    char **p_value_string;
} XHELLO_PARAMS;

/* Default parameter values (as strings) */
char *mbgcolor      = DEFAULT_BGCOLOR,
     *mfgcolor      = DEFAULT_FGCOLOR,
     *mfont         = DEFAULT_FONT,
     *display_name = NULL,
     *mtext         = DEFAULT_MAIN_TEXT,
     *mgeom         = NULL,
     *pre_text      = "Preprocess" ,
     *reg_text      = "Region" ,
     *mat_text      = "Material",
     *mesh_text     = "Mesh Info.",
     *quit_text     = "Quit",
     *cast_text     = "Cast",
     *mold_text     = "Mold",
     *steel_text    = "Steel",
     *al_text       = "Aluminum",
     *x_text        = "No. of Division (in X):",
     *y_text        = "No. of Division (in Y):",
     *cast_region   = " < Cast Region >",
     *mold_region   = " < Mold Region >";

/* List of user-customizable resources */
XHELLO_PARAMS resources[] =
{
    "font"              , &mfont
};
int num_resources = sizeof(resources) / sizeof(XHELLO_PARAMS);

char *app_name = "xpre1";
/* Nonstring parameters of xhello */
XFontStruct    *mfontstruct;
unsigned long  mbgpix, mfgpix,mbgpix1,mbgpix2,mbgpix3,mbgpix4;
/* Other global variables */
XWMHints    xwmh;           /* Hints for the window manager  */
XSizeHints  xsh;            /* Size hints for window manager */
Display     *p_disp;        /* Connection to X display       */
```

```
Window      theMain,
            theDis,
            thePre,
            theReg,
            theMat,
            theMesh,
            theQuit,
            theCast,
            theMold,
            theSteel,
            theAl,
            theX,
            theY   ;     /* Window ID of the two windows  */
GC          theGC,
            theGC1,
            theGC2;       /* Graphic context for main      */
XEvent      theEvent;    /* Structure for current event   */
int         Done = 0;    /* Flag to indicate wheen done   */
char        default_geometry[80];

int px=6, py=6, pxtxt, pytxt,
    disx, disy;

int rx,ry, mx,my, mex,mey, qx,qy, cx,cy, mlx,mly, sx,sy, ax,ay, xx,xy, yx
y;
unsigned int pwidth, pheight,
             dwidth, dheight,
             cwidth, cheight,
             swidth, sheight,
             xwidth, xheight;

int main_size_x = 1100;
int main_size_y = 900;

int region_flag,
    mat_flag;

struct
{
   short x,y;
} points[5];
struct
{
   float x,y;
} node[NMAX+1];
struct
{
    short min_px,max_px;
    short min_py,max_py;
    int division_x;
    int division_y;
    struct
    {
       int node_num;
       float x;
```

```
        float y;
        int px;
        int py;
      } node1, node2,node3,node4;
} blocks[NMAX + 1];

int num_node;
int num_block;
int point_x;
int point_y;
int blockpoint;

char *filestring1 = "cad1.dxf";
char *filestring2 = "cad2.dxf";

void cad(char *);
void usage(void);
/*-------------------------------------------------------------
main(int argc, char **argv)
{
    int i,n1,n2,n3,n4,nou=4,gabbage;
    float nothing=1.0;
    char                    *tmpstr;
    Colormap                default_cmap;
    XColor                  color;
    XGCValues               gcv;
    XSetWindowAttributes    xswa;
    char                    keystring[10];
    char                    totalx[20];
    char                    totaly[20];
    char                    *str;
    int                     nkey = 10;
    int                     count,key_value;
    KeySym                  key;
    XComposeStatus           cs;

    FILE *in, *out;

/* STEP 1:  Parse command line. Each of xhello's options
 *          require a value. So we process the command-line
 *          arguements in pairs.
 */

/* Step 2: Open connection to display. */
    if ((p_disp = XOpenDisplay(display_name)) == NULL)
    {
        fprintf(stderr, "%s: can't open display named %s\n",
                argv[0], XDisplayName(display_name));
        exit(1);
    }
/* Step 3: Get resources from the reource file. */
    for (i = 0; i < num_resources; i++)
    {
        if((tmpstr = XGetDefault(p_disp, app_name,
                                 resources[i].name)) != NULL)
```

```
            *resources[i].p_value_string = tmpstr;
    }
/* Step 4: Set up colors and fonts. */
/*         First the fonts....     */
    if ((mfontstruct = XLoadQueryFont(p_disp, mfont)) == NULL)
    {
        fprintf(stderr, "%s: display %s cannot load font %s\n",
                app_name, DisplayString(p_disp), mfont);
        exit(1);
    }
/* Now select the colors using the default colormap. */
    default_cmap = DefaultColormap(p_disp,
                                   DefaultScreen(p_disp));
/* Main window's background color */
    if (XParseColor(p_disp, default_cmap, mbgcolor,
                    &color) == 0 ||
        XAllocColor(p_disp, default_cmap, &color) == 0)
/* Use white foreground in case of failure. */
        mbgpix = WhitePixel(p_disp, DefaultScreen(p_disp));
    else
        mbgpix = color.pixel;
/* Main window's foreground color */
    if (XParseColor(p_disp, default_cmap, mfgcolor,
                    &color) == 0 ||
        XAllocColor(p_disp, default_cmap, &color) == 0)
/* Use black foreground in case of failure. */
        mfgpix = BlackPixel(p_disp, DefaultScreen(p_disp));
    else
        mfgpix = color.pixel;

    if (XParseColor(p_disp, default_cmap, "DarkSlateGrey",
                    &color) == 0 ||
        XAllocColor(p_disp, default_cmap, &color) == 0)
/* Use white foreground in case of failure. */
        mbgpix1 = WhitePixel(p_disp, DefaultScreen(p_disp));
    else
        mbgpix1 = color.pixel;

    if (XParseColor(p_disp, default_cmap, "DarkSeaGreen",
                    &color) == 0 ||
        XAllocColor(p_disp, default_cmap, &color) == 0)
/* Use white foreground in case of failure. */
        mbgpix2 = WhitePixel(p_disp, DefaultScreen(p_disp));
    else
        mbgpix2 = color.pixel;

    if (XParseColor(p_disp, default_cmap, "yellow2",
                    &color) == 0 ||
        XAllocColor(p_disp, default_cmap, &color) == 0)
/* Use white foreground in case of failure. */
        mbgpix3 = WhitePixel(p_disp, DefaultScreen(p_disp));
    else
        mbgpix3 = color.pixel;

    if (XParseColor(p_disp, default_cmap, "DodgerBlue1",
```

```
                    &color) == 0 ||
        XAllocColor(p_disp, default_cmap, &color) == 0)
/* Use white foreground in case of failure. */
        mbgpix4 = WhitePixel(p_disp, DefaultScreen(p_disp));
    else
        mbgpix4 = color.pixel;

/* STEP 5: Select initial position and size of top window. */
/* Fill out an XsizeHints structure to inform the window manager.
 * Here we pick a default size large enough to fit the text and
 * the Exit button.
*/
   pxtxt = mfontstruct->max_bounds.width / 2;
   pytxt = mfontstruct->max_bounds.ascent +
           mfontstruct->max_bounds.descent;
   pwidth = pxtxt + XTextWidth(mfontstruct, pre_text,strlen(pre_text))+2
   pheight = pytxt + 10;

   cwidth = pxtxt + XTextWidth(mfontstruct, cast_text,strlen(cast_text))
   cheight = pytxt + 6;

   swidth = pxtxt + XTextWidth(mfontstruct, al_text,strlen(al_text));
   sheight = pytxt + 6;

   xwidth = pxtxt + XTextWidth(mfontstruct, x_text,strlen(x_text))+60;
   xheight = pytxt + 6;


    xsh.flags = (PPosition |PSize );
    xsh.height = main_size_y;
    xsh.width = main_size_x;
    xsh.x = 200;
    xsh.y = 200;
/* Construct a default geometry string. */
    sprintf(default_geometry, "%dx%d+%d+%d", xsh.width,
            xsh.height, xsh.x, xsh.y);
    mgeom = default_geometry;

/* Override the geomtery, if necessary. */

/* Process the geometry specification. */

/* Check bitmask and set flags in XSizeHints structure. */

/* STEP 6: Create top-level window.    */
/* Use position and size information derived above.
 * For border color, use the foreground color.
 */
   theMain = XCreateSimpleWindow(p_disp, DefaultRootWindow(p_disp),
                  xsh.x, xsh.y, xsh.width, xsh.height,
                 DEFAULT_BDWIDTH, mfgpix, mbgpix1);
/* STEP 7: Set Window Manager properties (window name, icon
 * name, command-line, and size hints).
 */
  XSetStandardProperties(p_disp, theMain, app_name, app_name,
```

```
                                None, argv, argc, &xsh);
/* STEP 8: Give other hints to window manager. */
/* Fill the XWMHints structure and call XSetWMHints() */
    xwmh.flags = (InputHint|StateHint);
    xwmh.input = True;
    xwmh.initial_state = NormalState;
    XSetWMHints(p_disp, theMain, &xwmh);
/* STEP 9: Create a graphics context for the Main window. */
    gcv.font = mfontstruct->fid;
    gcv.foreground = mfgpix;
    gcv.background = mbgpix;
    gcv.line_width = 2;
    theGC = XCreateGC(p_disp, theMain,
                (GCFont | GCForeground | GCBackground|GCLineWidth), &gcv)

    gcv.foreground = mbgpix;
    gcv.background = mfgpix;
    theGC1 = XCreateGC(p_disp, theMain,
                     ( GCForeground | GCBackground), &gcv);

    gcv.foreground = mbgpix3;
    gcv.background = mfgpix;
    theGC2 = XCreateGC(p_disp, theMain,
                     ( GCForeground | GCBackground), &gcv);

/* STEP 10: Set window attributes (colormap, bit_gravity). */
    xswa.colormap = DefaultColormap(p_disp,
                                    DefaultScreen(p_disp));
    xswa.bit_gravity = CenterGravity;
    XChangeWindowAttributes(p_disp, theMain, (CWColormap |
                                CWBitGravity), &xswa);
/* STEP 11: Select input events for the Main window. */
    XSelectInput(p_disp, theMain, ExposureMask);
/* STEP 12: Map the Main window--to make it visible. */
    XMapWindow(p_disp, theMain);
/* STEP 13: Create any child windows, if any.
 * Also, select input events,  create graphics contexts
 * and map the child window.
 */

    disx = px + pwidth + 15;
    disy = py;
    dwidth = main_size_x - disx - 9;
    dheight = main_size_y - disy - 9;
    theDis = XCreateSimpleWindow(p_disp, theMain, disx, disy, dwidth, dhe
ht,
                                DEFAULT_BDWIDTH, mfgpix, mbgpix2);
XSelectInput(p_disp, theDis, ExposureMask|ButtonPressMask);
XMapWindow(p_disp,theDis);


thePre = XCreateSimpleWindow(p_disp, theMain, px, py,
                                pwidth, pheight, DEFAULT_BDWIDTH,
                                mfgpix, mbgpix3);
XSelectInput(p_disp, thePre, ExposureMask|ButtonPressMask);
```

```
    XMapWindow(p_disp,thePre);

    rx = px;
    ry = py + pheight + 6;
    theReg = XCreateSimpleWindow(p_disp, theMain, rx, ry,
                                 pwidth, pheight, DEFAULT_BDWIDTH,
                                 mfgpix, mbgpix);
    XSelectInput(p_disp, theReg, ExposureMask|ButtonPressMask);
    XMapWindow(p_disp,theReg);

    mx = px;
    my = ry + pheight + 3;
    theMat = XCreateSimpleWindow(p_disp, theMain, mx, my,
                                 pwidth, pheight, DEFAULT_BDWIDTH,
                                 mfgpix, mbgpix);
    XSelectInput(p_disp, theMat, ExposureMask|ButtonPressMask);
    XMapWindow(p_disp,theMat);

    mex = px;
    mey = my + pheight + 3;
    theMesh = XCreateSimpleWindow(p_disp, theMain, mex, mey,
                                 pwidth, pheight, DEFAULT_BDWIDTH,
                                 mfgpix, mbgpix);
    XSelectInput(p_disp, theMesh, ExposureMask|ButtonPressMask);
    XMapWindow(p_disp,theMesh);

    qx = px;
    qy = mey + pheight + 3;
    theQuit = XCreateSimpleWindow(p_disp, theMain, qx, qy,
                                 pwidth, pheight, DEFAULT_BDWIDTH,
                                 mfgpix, mbgpix);
    XSelectInput(p_disp, theQuit, ExposureMask|ButtonPressMask);
    XMapWindow(p_disp,theQuit);

    cx = rx + pwidth +3;
    cy = ry ;
    theCast = XCreateSimpleWindow(p_disp, theMain, cx, cy,
                                 cwidth, cheight, DEFAULT_BDWIDTH,
                                 mfgpix, mbgpix4);
    XSelectInput(p_disp, theCast, ExposureMask|ButtonPressMask);
    XMapWindow(p_disp,theCast);

    mlx = cx;
    mly = ry + cheight + 3;
    theMold = XCreateSimpleWindow(p_disp, theMain, mlx, mly,
                                 cwidth, cheight, DEFAULT_BDWIDTH,
                                 mfgpix, mbgpix4);
    XSelectInput(p_disp, theMold, ExposureMask|ButtonPressMask);
    XMapWindow(p_disp,theMold);

    sx = cx;
    sy = my;
    theSteel = XCreateSimpleWindow(p_disp, theMain, sx, sy,
                                 swidth, sheight, DEFAULT_BDWIDTH,
                                 mfgpix, mbgpix4);
```

```
    XSelectInput(p_disp, theSteel, ExposureMask|ButtonPressMask);
    XMapWindow(p_disp,theSteel);

    ax = sx;
    ay = my + sheight + 3;
    theAl = XCreateSimpleWindow(p_disp, theMain, ax, ay,
                                swidth, sheight, DEFAULT_BDWIDTH,
                                mfgpix, mbgpix4);
    XSelectInput(p_disp, theAl, ExposureMask|ButtonPressMask);
    XMapWindow(p_disp,theAl);

    xx = mex + pwidth + 3;
    xy = mey;
    theX = XCreateSimpleWindow(p_disp, theMain, xx, xy,
                               xwidth, xheight, DEFAULT_BDWIDTH,
                               mfgpix, mbgpix4);
    XSelectInput(p_disp, theX, ExposureMask|ButtonPressMask|KeyPressMask)
    XMapWindow(p_disp,theX);

    yx = xx;
    yy = mey + xheight + 3;
    theY = XCreateSimpleWindow(p_disp, theMain, yx, yy,
                               xwidth, xheight, DEFAULT_BDWIDTH,
                               mfgpix, mbgpix4);
    XSelectInput(p_disp, theY, ExposureMask|ButtonPressMask|KeyPressMask)
    XMapWindow(p_disp,theY);


    XUnmapWindow(p_disp,theCast);
    XUnmapWindow(p_disp,theMold);
    XUnmapWindow(p_disp,theSteel);
    XUnmapWindow(p_disp,theAl);
    XUnmapWindow(p_disp,theX);
    XUnmapWindow(p_disp,theY);


/* STEP 14: Retrieve and process events until done. */


    while(!Done)
    {
       XNextEvent(p_disp, &theEvent);
       if(theEvent.xany.window == theMain)
       {
       };

if(theEvent.xany.window == thePre)
   switch(theEvent.type)
   {
      case Expose:
           if (theEvent.xexpose.count == 0)
           {
               XClearWindow(p_disp, thePre);
               XDrawString(p_disp, thePre, theGC, pxtxt,
                     pytxt, pre_text, strlen(pre_text));
```

```
                }
                break;
         };
      if(theEvent.xany.window == theReg)
         switch(theEvent.type)
         {
            case Expose:
                 if (theEvent.xexpose.count == 0)
                 {
                     XClearWindow(p_disp, theReg);
                     XDrawString(p_disp, theReg, theGC, pxtxt,
                          pytxt, reg_text, strlen(reg_text));
                 }
                 break;
          case ButtonPress:
                 XMapRaised(p_disp,theCast);
                 XClearWindow(p_disp, theCast);
                 XDrawString(p_disp, theCast, theGC, pxtxt,
                              pytxt, cast_text, strlen(cast_text));
                 XMapRaised(p_disp,theMold);
                 XClearWindow(p_disp, theMold);
                 XDrawString(p_disp, theMold, theGC, pxtxt,
                              pytxt, mold_text, strlen(mold_text));
                 break;
        };


    if(theEvent.xany.window == theCast)
    switch(theEvent.type)
    {
       case Expose:
            if (theEvent.xexpose.count == 0)
            {
                XClearWindow(p_disp, theCast);
                XDrawString(p_disp, theCast, theGC, pxtxt,
                     pytxt, cast_text, strlen(cast_text));
            }
            break;
      case ButtonPress:
            region_flag = 2;
            cad(filestring2);
            in = fopen("cad.out","rt");
            fscanf(in,"%d\n",&num_block);
            fscanf(in,"%d\n",&num_node);
            for(i=1;i<=num_node;i++)
            {
               fscanf(in,"%f %f\n",&node[i].x,&node[i].y);
            };
            for(i=1;i<= num_block;i++)
            {
             fscanf(in,"%d %d %d %d %d\n",&gabbage,&blocks[i].node1
ode_num,
                    &blocks[i].node2.node_num, &blocks[i].node3.nod
num,
                    &blocks[i].node4.node_num);
```

```
};
fclose(in);
for(i = 1;i <= num_block; i++)
{
    n1 = blocks[i].node1.node_num;
    n2 = blocks[i].node2.node_num;
    n3 = blocks[i].node3.node_num;
    n4 = blocks[i].node4.node_num;

    blocks[i].node1.x = node[n1].x;
    blocks[i].node1.y = node[n1].y;

    blocks[i].node2.x = node[n2].x;
    blocks[i].node2.y = node[n2].y;

    blocks[i].node3.x = node[n3].x;
    blocks[i].node3.y = node[n3].y;

    blocks[i].node4.x = node[n4].x;
    blocks[i].node4.y = node[n4].y;

    blocks[i].node1.px = (blocks[i].node1.x) * 40 + 250
    blocks[i].node1.py = 710 - (blocks[i].node1.y) * 40
    blocks[i].node2.px = (blocks[i].node2.x) * 40 + 250
    blocks[i].node2.py = 710 - (blocks[i].node2.y) * 40
    blocks[i].node3.px = (blocks[i].node3.x) * 40 + 250
    blocks[i].node3.py = 710 - (blocks[i].node3.y) * 40
    blocks[i].node4.px = (blocks[i].node4.x) * 40 + 250
    blocks[i].node4.py = 710 - (blocks[i].node4.y) * 40
};
for(i=1;i<=num_block;i++)
{
    blocks[i].min_px=blocks[i].node1.px;
    blocks[i].max_px=blocks[i].node1.px;
    blocks[i].min_py=blocks[i].node1.py;
    blocks[i].max_py=blocks[i].node1.py;

    if(blocks[i].min_px > blocks[i].node2.px)
       blocks[i].min_px = blocks[i].node2.px;
    if(blocks[i].min_py > blocks[i].node2.py)
       blocks[i].min_py = blocks[i].node2.py;
    if(blocks[i].max_px < blocks[i].node2.px)
       blocks[i].max_px = blocks[i].node2.px;
    if(blocks[i].max_py < blocks[i].node2.py)
       blocks[i].max_py = blocks[i].node2.py;

if(blocks[i].min_px > blocks[i].node3.px)
   blocks[i].min_px = blocks[i].node3.px;
if(blocks[i].min_py > blocks[i].node3.py)
   blocks[i].min_py = blocks[i].node3.py;
if(blocks[i].max_px < blocks[i].node3.px)
   blocks[i].max_px = blocks[i].node3.px;
if(blocks[i].max_py < blocks[i].node3.py)
   blocks[i].max_py = blocks[i].node3.py;
```

```
      if(blocks[i].min_px > blocks[i].node4.px)
         blocks[i].min_px = blocks[i].node4.px;
      if(blocks[i].min_py > blocks[i].node4.py)
         blocks[i].min_py = blocks[i].node4.py;
      if(blocks[i].max_px < blocks[i].node4.px)
         blocks[i].max_px = blocks[i].node4.px;
      if(blocks[i].max_py < blocks[i].node4.py)
         blocks[i].max_py = blocks[i].node4.py;
};
            XUnmapWindow(p_disp, theCast);
            XUnmapWindow(p_disp, theMold);

     };

if(theEvent.xany.window == theMold)
   switch(theEvent.type)
   {
      case Expose:
            if (theEvent.xexpose.count == 0)
            {
                XClearWindow(p_disp, theMold);
                XDrawString(p_disp, theMold, theGC, pxtxt,
                      pytxt, mold_text, strlen(mold_text));
            }
            break;
     case ButtonPress:
            region_flag = 1;
            /* cad(filestring1); */
            in = fopen("cad.out","rt");
            fscanf(in,"%d\n",&num_block);
            fscanf(in,"%d\n",&num_node);
            for(i=1;i<=num_node;i++)
            {
               fscanf(in,"%f %f\n",&node[i].x,&node[i].y);
            };
            for(i=1;i<= num_block;i++)
            {
             fscanf(in,"%d %d %d %d %d\n",&gabbage,&blocks[i].node1
im,
                   &blocks[i].node2.node_num, &blocks[i].node3.nod

                   &blocks[i].node4.node_num);
            };
            fclose(in);
            for(i = 1;i <= num_block; i++)
            {
            n1 = blocks[i].node1.node_num;
            n2 = blocks[i].node2.node_num;
            n3 = blocks[i].node3.node_num;
            n4 = blocks[i].node4.node_num;

            blocks[i].node1.x = node[n1].x;
            blocks[i].node1.y = node[n1].y;

            blocks[i].node2.x = node[n2].x;
```

```
        blocks[i].node2.y = node[n2].y;

        blocks[i].node3.x = node[n3].x;
        blocks[i].node3.y = node[n3].y;

        blocks[i].node4.x = node[n4].x;
        blocks[i].node4.y = node[n4].y;

        blocks[i].node1.px = (blocks[i].node1.x) * 40 + 250
        blocks[i].node1.py = 710 - (blocks[i].node1.y) * 40
        blocks[i].node2.px = (blocks[i].node2.x) * 40 + 250
        blocks[i].node2.py = 710 - (blocks[i].node2.y) * 40
        blocks[i].node3.px = (blocks[i].node3.x) * 40 + 250
        blocks[i].node3.py = 710 - (blocks[i].node3.y) * 40
        blocks[i].node4.px = (blocks[i].node4.x) * 40 + 250
        blocks[i].node4.py = 710 - (blocks[i].node4.y) * 40
    };

    for(i=1;i<=num_block;i++)
    {
        blocks[i].min_px=blocks[i].node1.px ;
        blocks[i].max_px=blocks[i].node1.px ;
        blocks[i].min_py=blocks[i].node1.py ;
        blocks[i].max_py=blocks[i].node1.py ;

        if(blocks[i].min_px > blocks[i].node2.px)
           blocks[i].min_px = blocks[i].node2.px;
        if(blocks[i].min_py > blocks[i].node2.py)
           blocks[i].min_py = blocks[i].node2.py;
        if(blocks[i].max_px < blocks[i].node2.px)
           blocks[i].max_px = blocks[i].node2.px;
        if(blocks[i].max_py < blocks[i].node2.py)
           blocks[i].max_py = blocks[i].node2.py;

        if(blocks[i].min_px > blocks[i].node3.px)
           blocks[i].min_px = blocks[i].node3.px;
        if(blocks[i].min_py > blocks[i].node3.py)
           blocks[i].min_py = blocks[i].node3.py;
        if(blocks[i].max_px < blocks[i].node3.px)
           blocks[i].max_px = blocks[i].node3.px;
        if(blocks[i].max_py < blocks[i].node3.py)
           blocks[i].max_py = blocks[i].node3.py;

        if(blocks[i].min_px > blocks[i].node4.px)
           blocks[i].min_px = blocks[i].node4.px;
        if(blocks[i].min_py > blocks[i].node4.py)
           blocks[i].min_py = blocks[i].node4.py;
        if(blocks[i].max_px < blocks[i].node4.px)
           blocks[i].max_px = blocks[i].node4.px;
        if(blocks[i].max_py < blocks[i].node4.py)
           blocks[i].max_py = blocks[i].node4.py;
    };

    XUnmapWindow(p_disp, theCast);
    XUnmapWindow(p_disp, theMold);
```

```
        };

    if(theEvent.xany.window == theMat)
       switch(theEvent.type)
       {
          case Expose:
               if (theEvent.xexpose.count == 0)
               {
                   XClearWindow(p_disp, theMat);
                   XDrawString(p_disp, theMat, theGC, pxtxt,
                              pytxt, mat_text, strlen(mat_text));
               }
               break;
         case ButtonPress:
               XMapRaised(p_disp,theSteel);
               XClearWindow(p_disp, theSteel);
               XDrawString(p_disp, theSteel, theGC, pxtxt,
                           pytxt, steel_text, strlen(steel_text));
               XMapRaised(p_disp,theAl);
               XClearWindow(p_disp, theAl);
               XDrawString(p_disp, theAl, theGC, pxtxt,
                           pytxt, al_text, strlen(al_text));
               break;

        };
    if(theEvent.xany.window == theSteel)
       switch(theEvent.type)
       {
         case Expose:
               if (theEvent.xexpose.count == 0)
               {
                   XClearWindow(p_disp, theSteel);
                   XDrawString(p_disp, theSteel, theGC, pxtxt,
                        pytxt, steel_text, strlen(steel_text));
               }
               break;
         case ButtonPress:
               mat_flag = 1;
               XUnmapWindow(p_disp, theSteel);
               XUnmapWindow(p_disp, theAl);
               break;
        };

    if(theEvent.xany.window == theAl)
       switch(theEvent.type)
       {
         case Expose:
               if (theEvent.xexpose.count == 0)
               {
                   XClearWindow(p_disp, theAl);
                   XDrawString(p_disp, theAl, theGC, pxtxt,
                        pytxt, al_text, strlen(al_text));
               }
               break;
    case ButtonPress:
```

```
                mat_flag = 2;
                XUnmapWindow(p_disp, theSteel);
                XUnmapWindow(p_disp, theAl);
                break;
            };

        if(theEvent.xany.window == theDis)
           switch(theEvent.type)
           {
              case Expose:
                   if (theEvent.xexpose.count == 0)
{
    XClearWindow(p_disp, theDis);
    for(i=1;i<=num_block;i++)
    {
      points[0].x=blocks[i].node1.px;
      points[0].y=blocks[i].node1.py;
      points[1].x=blocks[i].node2.px;
      points[1].y=blocks[i].node2.py;
      points[2].x=blocks[i].node3.px;
      points[2].y=blocks[i].node3.py;
      points[3].x=blocks[i].node4.px;
      points[3].y=blocks[i].node4.py;


      XFillPolygon(p_disp, theDis,
                   theGC1, points, 4,
                   Convex, CoordModeOrigin);
    };

    for(i=1;i<=num_block;i++)
    {
      points[0].x=blocks[i].node1.px;
      points[0].y=blocks[i].node1.py;
      points[1].x=blocks[i].node2.px;
      points[1].y=blocks[i].node2.py;
      points[2].x=blocks[i].node3.px;
      points[2].y=blocks[i].node3.py;
      points[3].x=blocks[i].node4.px;
      points[3].y=blocks[i].node4.py;
      points[4].x=blocks[i].node1.px;
      points[4].y=blocks[i].node1.py;

      XDrawLines(p_disp, theDis, theGC,
                 points, 5, CoordModeOrigin);
    };
    if(region_flag ==1)
    {
       XDrawString(p_disp, theDis, theGC, 400,
                   750, mold_region, strlen(mold_region
;
    }
    if(region_flag ==2)
    {
```

```
                        XDrawString(p_disp, theDis, theGC, 400,
                                    750, cast_region, strlen(cast_region
;
                    }
                }
                break;
            case ButtonPress:
                point_x = theEvent.xbutton.x;
                point_y = theEvent.xbutton.y;
                blockpoint = 0;
                for(i=1;i<=num_block;i++)
                {
                  if((point_x > blocks[i].min_px) && (point_y > blocks[
.min_py) &&
                     (point_x < blocks[i].max_px) && (point_y < blocks[
.max_py))
                     blockpoint = i;
                }

                if(blockpoint)
                {
                points[0].x=blocks[blockpoint].node1.px;
                points[0].y=blocks[blockpoint].node1.py;
                points[1].x=blocks[blockpoint].node2.px;
                points[1].y=blocks[blockpoint].node2.py;
                points[2].x=blocks[blockpoint].node3.px;
                points[2].y=blocks[blockpoint].node3.py;
                points[3].x=blocks[blockpoint].node4.px;
                points[3].y=blocks[blockpoint].node4.py;

                XFillPolygon(p_disp, theDis,
                             theGC2, points, 4,
                             Convex, CoordModeOrigin);
                }
                for(i=1;i<=num_block;i++)
                {
                  points[0].x=blocks[i].node1.px;
                  points[0].y=blocks[i].node1.py;
                  points[1].x=blocks[i].node2.px;
                  points[1].y=blocks[i].node2.py;
                  points[2].x=blocks[i].node3.px;
                  points[2].y=blocks[i].node3.py;
                  points[3].x=blocks[i].node4.px;
                  points[3].y=blocks[i].node4.py;
                  points[4].x=blocks[i].node1.px;
                  points[4].y=blocks[i].node1.py;

                    XDrawLines(p_disp, theDis, theGC,
                               points, 5, CoordModeOrigin);

                };

                break;
};
```

```
        if(theEvent.xany.window == theMesh)
           switch(theEvent.type)
           {
              case Expose:
                   if (theEvent.xexpose.count == 0)
                   {
                       XClearWindow(p_disp, theMesh);
                       XDrawString(p_disp, theMesh, theGC, pxtxt,
                            pytxt, mesh_text, strlen(mesh_text));
                   }
                   break;

             case ButtonPress:
                   XMapRaised(p_disp,theX);
                   XClearWindow(p_disp, theX);
                   XDrawString(p_disp, theX, theGC, pxtxt,
                               pytxt, x_text, strlen(x_text));
                   XMapRaised(p_disp,theY);
                   XClearWindow(p_disp, theY);
                   XDrawString(p_disp, theY, theGC, pxtxt,
                               pytxt, y_text, strlen(y_text));

                   totalx[0]='\0';
                   totaly[0]='\0';

                   break;
          };

        if(theEvent.xany.window == theX)
           switch(theEvent.type)
           {
              case Expose:
                   if (theEvent.xexpose.count == 0)
                   {
                       XClearWindow(p_disp, theX);
                       XDrawString(p_disp, theX, theGC, pxtxt,
                            pytxt, x_text, strlen(x_text));
                   }
                   break;
              case KeyPress:
                   count = XLookupString(&theEvent,keystring,nkey,&key,&cs
                   keystring[count] = '\0';
                   str = strcat( totalx, keystring);
                   key_value=atoi(str);
                   XClearWindow(p_disp, theX);
                   XDrawString(p_disp, theX, theGC, pxtxt,
                         pytxt, x_text, strlen(x_text));
                   XDrawString(p_disp, theX, theGC, (xwidth -30),
                               pytxt, str, strlen(str));
                   break;
         case ButtonPress:
                   blocks[blockpoint].division_x = key_value;
                   break;
        };
```

```
        if(theEvent.xany.window == theY)
           switch(theEvent.type)
           {
              case Expose:
                   if (theEvent.xexpose.count == 0)
                   {
                       XClearWindow(p_disp, theY);
                       XDrawString(p_disp, theY, theGC, pxtxt,
                             pytxt, y_text, strlen(y_text));
                   }
                   break;
            case KeyPress:
                   count = XLookupString(&theEvent,keystring,nkey,&key,&cs
                   keystring[count] = '\0';
                   str = strcat( totaly, keystring);
                   key_value=atoi(str);
                   XClearWindow(p_disp, theY);
                   XDrawString(p_disp, theY, theGC, pxtxt,
                         pytxt, y_text, strlen(y_text));
                   XDrawString(p_disp, theY, theGC, (xwidth -30),
                               pytxt, str, strlen(str));
                   break;
          case ButtonPress:
                   blocks[blockpoint].division_y = key_value;
                   XUnmapWindow(p_disp, theX);
                   XUnmapWindow(p_disp, theY);
                   break;
            };

        if(theEvent.xany.window == theQuit)
           switch(theEvent.type)
           {
              case Expose:
                   if (theEvent.xexpose.count == 0)
                   {
                       XClearWindow(p_disp, theQuit);
                       XDrawString(p_disp, theQuit, theGC, pxtxt,
                             pytxt, quit_text, strlen(quit_text));
                   }
                   break;
              case ButtonPress:
                   Done = 1;
                   break;
            } ;
};

out = fopen("xpre1.dat","wt");
fprintf(out,"    0    0\n");
fprintf(out,"%5d\n",num_block);
fprintf(out,"%5d\n",num_node);
for(i=1;i<=num_node;i++)
{
   fprintf(out,"%10.4f %10.4f\n",node[i].x, node[i].y);
};
```

```
    for(i=1;i<=num_block;i++)
    {
       fprintf(out,"%5d\n",mat_flag);
       fprintf(out,"%5d %4d\n", blocks[i].division_x,
                                blocks[i].division_y);
       fprintf(out,"%5d\n",nou);
       fprintf(out,"%5d\n", blocks[i].node1.node_num);
       fprintf(out,"%5d\n", blocks[i].node2.node_num);
       fprintf(out,"%5d\n", blocks[i].node3.node_num);
       fprintf(out,"%5d\n", blocks[i].node4.node_num);
       fprintf(out,"%10.1f %10.1f\n", nothing, nothing);
    };
    fprintf(out,"%s\n","NO");
    fprintf(out,"%s\n","NO");
    fclose(out);


/* STEP 15: Close connection to display and exit. */
    XFreeGC(p_disp, theGC);
    XDestroyWindow(p_disp, theMain);
    XCloseDisplay(p_disp);
    exit(0);
};
/*-------------------------------------------------------------*
void usage(void)
{
     fprintf (stderr, "usage: %s [-display host:display] \
[-geometry geom ] [-mtext text] [-etext text]\n", app_name);
     exit(1);
}


#define   LMAX  30          /* the number of maximum lines */
int    possible() ;
int    n , nn , nb ;
int    node_st[NMAX][7] ;
static int   vect[NMAX][NMAX] ;
float  from_x[LMAX] ,from_y[LMAX], to_x[LMAX], to_y[LMAX] ;
float  node_x[NMAX], node_y[NMAX] ;
int    block[NMAX][4] ;

void cad(char *filestring)
{
int    i , j , k , l , p ;
int    neig , conn[LMAX] , poss[3] ;
float  up1_x, up2_x, lo1_x, lo2_x, m1, m2, x, y ;
float  up1_y, up2_y, lo1_y, lo2_y ;
float  min, max, vec_x, vec_y, mag[LMAX] ;
char   string[21] ,tmp[21], cond, cond2 ;
char   data_file[20] ,output_file[20] ;
FILE   *fi ,*fo ;

if ((fi = fopen(filestring,"rt")) == NULL)
        {
        printf("File \"%s\" does not exist.\n",data_file);
```

```
        exit(0);
        }


/*  reading data file. reading only lines.  */
n = 0 ;   /* n is the number of lines */
while (!feof(fi))
        {
        fscanf(fi,"%s",string) ;
        if (!strcmp(string ,"LINE"))
                {
                for(i = 1 ; i <= 3 ; i++)
                        fscanf(fi,"%s",tmp);
                fscanf(fi,"%f",&from_x[n]) ;
                        fscanf(fi,"%s",tmp);
                fscanf(fi,"%f",&from_y[n]) ;
                for(i = 1 ; i <= 3 ; i++)
                        fscanf(fi,"%s",tmp);
                fscanf(fi,"%f",&to_x[n]) ;
                        fscanf(fi,"%s",tmp);
                fscanf(fi,"%f",&to_y[n]) ;
                n++ ;
                }
        }

fclose(fi) ;

/*  generating nodes from intersection of two combination of lines  */
nn = 0 ;   /* nn is the number of nodes */
for (i = 0 ; i < n ; i++)
    for (j = i+1 ; j < n ; j++)
            {
            cond2 = 'n' ;
            /* calculating intersection point */
            if ((to_x[i] == from_x[i]) && (to_x[j] != from_x[j]))
                {
                x = to_x[i] ;
                m2 = (to_y[j] - from_y[j])/(to_x[j] - from_x[j]) ;
                y = m2*(x - from_x[j]) + from_y[j] ;
                cond2 = 'y' ;
                }
            else if ((to_x[i] != from_x[i]) && (to_x[j] == from_x[j]))
                {
                x = to_x[j] ;
                m1 = (to_y[i] - from_y[i])/(to_x[i] - from_x[i]) ;
                y = m1*(x - from_x[i]) + from_y[i] ;
                cond2 = 'y' ;
                }
           else if ((to_x[i] != from_x[i]) && (to_x[j] != from_x[j]))
                {
                m1 = (to_y[i] - from_y[i])/(to_x[i] - from_x[i]) ;
                m2 = (to_y[j] - from_y[j])/(to_x[j] - from_x[j]) ;
                if (m1 != m2)
                    {
```

```
                    x = (from_y[j] - from_y[i] + m1*from_x[i] - m2*from_x
])
                       /(m1 - m2) ;
                    y = from_y[i] + m1*(x - from_x[i]) ;
                    cond2 = 'y' ;
                    }
                }

            if (cond2 == 'y')
                {
                up1_x = (from_x[i] > to_x[i]) ? from_x[i] : to_x[i] ;
                lo1_x = (from_x[i] < to_x[i]) ? from_x[i] : to_x[i] ;
                up2_x = (from_x[j] > to_x[j]) ? from_x[j] : to_x[j] ;
                lo2_x = (from_x[j] < to_x[j]) ? from_x[j] : to_x[j] ;
                up1_y = (from_y[i] > to_y[i]) ? from_y[i] : to_y[i] ;
                lo1_y = (from_y[i] < to_y[i]) ? from_y[i] : to_y[i] ;
                up2_y = (from_y[j] > to_y[j]) ? from_y[j] : to_y[j] ;
                lo2_y = (from_y[j] < to_y[j]) ? from_y[j] : to_y[j] ;

                /*  inspecting whether the intersection point is
                    within  proper range  */
                if ( (x <= up1_x && x >= lo1_x) && (x <= up2_x && x >= lo
x)
                  && (y <= up1_y && y >= lo1_y) && (y <= up2_y && y >= lo
y) )
                    {
                    /*  node generation  */
                    node_x[nn] = x ;
                    node_y[nn] = y ;
                    /*  storing status of node  */
                    node_st[nn][0] = 2 ;    /* composed by two lines */
                    node_st[nn][1] = i ;    /* first line number */
                    node_st[nn][2] = j ;    /* second line number */

                    /*  inspecting whether the nodes duplicate  */
                    for (k = 0 ; k < nn ; k++)
                       if(node_x[k] == node_x[nn] && node_y[k] == node_y[
])
                            /* the case of duplication */
                            {
                            cond = 'y' ;
                            for (l = 1 ; l <= node_st[k][0] ; l++)
                                if (node_st[nn][1] == node_st[k][l])
                                        cond = 'n' ;
                            if (cond == 'y')
                                /* storing of additional line */
                                {
                                node_st[k][0] = node_st[k][0] + 1 ;
                                node_st[k][node_st[k][0]] = node_st[nn][1
;
                                }

                            cond = 'y' ;
                            for (l = 1 ; l <= node_st[k][0] ; l++)
                                if (node_st[nn][2] == node_st[k][l])
```

```
                                cond = 'n' ;
                        if (cond == 'y')
                            /* storing of additional line */
                            {
                            node_st[k][0] = node_st[k][0] + 1 ;
                            node_st[k][node_st[k][0]] = node_st[nn][2
;
                            }
                        nn-- ;
                        break ;
                                }
                            nn++ ;
                            }
                    }
                }

/*  derivation of nodes which connect together  */
for (i = 0 ; i < nn ; i++)
        for (j = 1 ; j <= node_st[i][0] ; j++)
                {
                /*  derivation of nodes which are on the same line  */
                p = 0 ;
                for (k = 0 ; k < nn ; k++)
                    if (k != i)
                        for (l = 1 ; l <= node_st[k][0] ; l++)
                                if (node_st[k][l] == node_st[i][j])
{
conn[p] = k ;
vec_x = node_x[k] - node_x[i] ;
vec_y = node_y[k] - node_y[i] ;
mag[p] = vec_x*vec_x + vec_y*vec_
if (vec_x != 0.0)
            {
            if (vec_x < 0.0)
                    mag[p] = -mag[p]
            }
else
            {
            if (vec_y < 0.0)
                    mag[p] = -mag[p]
            }
p++ ;
break ;
}
                /*  selecting the nearest nodes  */
                cond = 'n' ;
                for (k = 0 ; k < p ; k++)
                    if (mag[k] > 0.0)
                            {
                            cond = 'y' ;
                            neig = conn[k] ; /* neig means neighbor n
e */
```

```
                        min = mag[k] ;
                        break ;
                        }
                if (cond == 'y')
                        {
                        for (l = k ; l < p ; l++)
                                if (mag[l] > 0.0 )
                                        if (mag[l] < min )
                                                {
                                                min = mag[l] ;
                                                neig = conn[l] ;
                                                }
                        /* i_th and neig_th nodes are connected together
                        vect[i][neig] = 1 ;
                        }

        cond = 'n' ;
        for (k = 0 ; k < p ; k++)
                if (mag[k] < 0.0)
                        {
                        cond = 'y' ;
                        neig = conn[k] ;
                        max = mag[k] ;
                        break ;
                        }
        if (cond == 'y')
                {
                for (l = k ; l < p ; l++)
                        if (mag[l] < 0.0 )
                                if (mag[l] > max )
                                        {
                                        max = mag[l] ;
                                        neig = conn[l] ;
                                        }
                /* i_th and neig_th nodes are connected together
                vect[i][neig] = 1 ;
                }
        }


/*  block generation  */
nb = 0 ;    /* nb is block number */
for (i = 0 ; i < nn ; i++)
        for (j = 0 ; j < nn ; j++)
                if (vect[i][j] == 1)
                        {
                        poss[0] = possible(i,j) ;
                        if (poss[0] != -1)
                        /* poss = -1 means that there are no line possibl
*/
                                {
                                poss[1] = possible(j,poss[0]) ;
                                if (poss[1] != -1)
```

```
                                        {
                                        poss[2] = possible(poss[0],poss[1
 ;
                                        if (poss[2] == i)
                                                {
                                                block[nb][0] = i ;
                                                block[nb][1] = j ;
                                                block[nb][2] = poss[0] ;
                                                block[nb][3] = poss[1] ;
                                                nb++ ;

                                                vect[i][j] = 2 ;
                                                vect[j][poss[0]] = 2 ;
                                                vect[poss[0]][poss[1]] =
;
                                                vect[poss[1]][i] = 2 ;
                                                }
                                        }
                                }
                        }


/*  output of the result  */
fo = fopen("cad.out","wt") ;
fprintf(fo,"%5d\n",nb);
fprintf(fo,"%5d\n",nn);
for (i = 0 ; i < nn ; i++)
    fprintf(fo,"%10.4f %10.4f\n", node_x[i], node_y[i]) ;

for (i = 0 ; i < nb ; i++)
    fprintf(fo,"%5d %5d %5d %5d %5d\n", i+1, block[i][0]+1,
            block[i][1]+1, block[i][2]+1, block[i][3]+1) ;

fclose(fo);

}


int  possible(i,j)     /* This function investigetes what vector is possib
 */
int  i , j ;           /* to compose block with the vector whose direction
 */
{                      /* is from i_th node to j_th node.
 */
int    k , poss ;
float  vec1_x, vec1_y, vec2_x, vec2_y, mag1, mag2, cross, inner, min, cos

vec1_x = node_x[j] - node_x[i] ;
vec1_y = node_y[j] - node_y[i] ;
mag1 = sqrt(vec1_x*vec1_x + vec1_y*vec1_y) ;  /* magnitude of vector */

poss = -1 ;
```

```
min = 1.0 ;
for (k = 0 ; k < nn ; k++)
        if (vect[j][k] == 1)
                {
                vec2_x = node_x[k] - node_x[j] ;
                vec2_y = node_y[k] - node_y[j] ;
                cross = vec1_x*vec2_y - vec1_y*vec2_x ;  /* cross product
/
                if (cross > 0.0)
                        /* we want counterclock_wise */
                        {
                        mag2 = sqrt(vec2_x*vec2_x +
                                vec2_y*vec2_y) ;
                        inner = vec1_x*vec2_x +          /* inner product
/
                                vec1_y*vec2_y  ;
                        cos = inner/(mag1*mag2) ;
                        if (cos < min)                   /* we must choose whose
/
                            {                            /* angle is the greater
/
                            min = cos ;
                            poss = k ;
                            }
                        }
                }

return(poss) ;  /* poss = -1 means that there are no line possible */
}
```

부록 2 : 자동격자생성 프로그램

(FORTRAN language 사용)

```
C*************************************************************************

C              PROGRAM : PRE-PROCESSOR

C                                                    DEPT. PRODUCTION ENG.
C                                                    KAIST
C                                                    1992. 5.
C*************************************************************************
C*************************************************************************

C    PURPOSE:
C       AUTOMETIC MESH GENERATOR FOR FINITE ELEMENT ANALYSIS USING
C       3-NODE TRIANGULAR ELEMENTS FOR PLANE AND AXISYSMETRIC
C       PROBLEMS IN MECHANICS

C    VARIABLE: < IMPORTANT >
C       NX      = TOTAL NUMBER OF NODAL POINTS
C       NELX    = TOTAL NUMBER OF FINITE ELEMENTS
C       IJK     = ELEMENT CONNECTIVITIES
C       X,Y     = COORDINATES OF NODAL POINTS
C       MPE     = THE GROUP NUMBER OF MATERIALS ASSIGNED EACH ELEMENT
C       NODE    = TOTAL NUMBER OF NODAL POINTS IN AN ELEMENT
C       NBX     = TOTAL NUMBER OF "NODES" THAT CHARACTERIZE BLOCKS
C       XB,YB   = COORDINATES OF "NODES" OF THE BLOCKS
C       XO,YO   = COORDINATES OF NODAL POINTS IN THE OLD NUMBERING SYSTEM
C       JOP1    = STORES THE NUMBER OF ELEMENTS CONNECTED TO A NODAL
C                 POINT
C       JOP2    = STORES ELEMENT NUMBERS CONNECTED TO A NODAL POINT
C       JOP3    = CONTAINS NODE NUMBERS IN THE OLD SYSTEM
C       JOP4    = DUMMY STORAGE
C       JNEW    = STORES NODE NUMBERS IN THE NEW NUMBERING SYSTEM
C       LLXY    = TOTAL NUMBER OF "NODES" IN A BLOCK ( 4,8 OR 12 )
C       IIJB    = BLOCK CONNECTIVITIES
C       MATPR   = GROUP NUMBER OF MATERIALS FOR THE BLOCK
C       MMX     = NUMBER OF DIVISIONS IN THE 1-ST DIRECTION
C       MMY     = NUMBER OF DIVISIONS IN THE 2-ND DIRECTION
C       PPX     = MESH GRADIENT IN THE 1-ST DIIRECTION
C       PPY     = MESH GRADIENT IN THE 2-ND DIIRECTION
C       LLMN    = NODE CONNECTIVITIES
C       LBLOCK  = TOTAL NUMBER OF BLOCKS USED FOR MESH GENERATION
C       LHANDR  = CONTROL NUMBER FOR MANUAL INPUT DATA
C       LR,LW   = CONTROL NUMBERS FOR READ AND WRITE STATEMENTS
C       LP      = CONTROL NUMBER TO STORE THE DATA

C    INPUT DATA TO BE READ:
C       1. MAIN ROUTINE
C          1.1 LBLOCK                              :    I5
C          IF(LBLOCK.EQ.0) GOTO THE 3-RD STEP
C          1.2 NBX                                 :    I5
C          1.3 XB(I),YB(I),I=1,NBX                 :    2F10.4
C       2. SUBROUTINE READBL
C          DO 100 I=1,LBLOCK
```

```
C          2.1 MATPR(N)                              :    I5
C          2.2 MMX(N),MMY(N)                         :    2I5
C          2.3 LLXY(N)                               :    I5
C          2.4 IIJB(I),I=1,LLXY(N)                   :    I5
C          2.5 PPX(N),PPY(N)                         :    2F10.4
C          100 CONTINUE
C       3. MAIN ROUTINE
C          3.1 LHANDR                                :    A4
C          IF(LHANDR.NE.'YES') GOTO THE 5-TH STEP
C       4. SUBROUTINE HRMESH
C          4.1 NODADD                                :    I5
C          4.2 NELADD                                :    I5
C          4.3 N.X(N),Y(N),I=1,NODADD                :    I5, 2F10.4
C          4.4 N,MPE(N),(IJK(J,N),J=1,NODE),
C              I=1,NELADD                            :    2I5,NODE*I5
C       5. MAIN ROUTINE
C          5.1 LOPTIM                                :    A4

C    DIMENSIONING :
C       X(NX),Y(NX),XO(NX),YO(NX)
C          NX   = TOTAL NUMBER OF NODAL POINTS CREATED BEFORE CONNECTION
C       IJK(NODE,NELX),MPE(NELX),P(3,NELX)
C          NODE = TOTAL NUMBER OF NODAL POINTS IN AN ELEMENT
C          NELX = TOTAL NUMBER OF ELEMENTS IN A MODEL
C       JOP1(NX),JOP2(8*NX),JOP3(NX),JOP4(NX),JNEW(NX)
C       XB(NBX),YB(NBX)
C          NBX  = TOTAL NUMBER OF "NODES" THAT CHARACTERIZE BLOCKS
C       MATPR(LBLOCK),MMX(LBLOCK),MMY(LBLOCK),PPX(LBLOCK),PPY(LBLOCK)
C        IIJB(LXY,LBLOCK),LLMN(4,LBLOCK),LLXY(LBLOCK)
C          LBLOCK = TOTAL NUMBERR OF BLOCKS
C          LXY    = TOTAL NUMBER OF "NODES" THAT DEFINE THE BLOCK
C          IJB(LXY)

C*************************************************************************

      DIMENSION X(1000),Y(1000),XO(1000),YO(1000),
     1          IJK(3,2000),MPE(2000),
     2          JOP1(1000),JOP2(8000),JOP3(1000),JOP4(1000),JNEW(1000),
     3          XB(400),YB(400),
     4          MATPR(400),MMX(400),MMY(400),PPX(400),PPY(400),
     5          IIJB(12,400),LLMN(4,400),LLXY(400),
     6          nbb(400),nbn(400)
      DIMENSION IJB(12)

      DATA LR,LW,LP,NODE,NDF/10,11,12,3,1/
      DATA IYES,INO/'YES','NO'/

      OPEN(10,FILE='pre.in')
      OPEN(11,FILE='pre.ref')
      OPEN(12,FILE='pre.dat')

C     ( READ CONTROL NUMBERS )

      read(lr,10) npnode,npelem
   10 format(2i5)
```

```
      WRITE(LW,500)
  500 FORMAT(//,'AUTOMETIC MESH GENERATOR')
      WRITE(LW,504)
  504 FORMAT(//,'HOW MANY BLOCKS ARE USED TO CREATE A FINITE EELEMENT',
     1  ' MODEL?',/,'INPUT THE NUMBER OF THE BLOCKS.')
      READ(LR,502) LBLOCK
  502 FORMAT(16I5)

C     ( READ THE COORDINATES OF "NODES" CHARACTERIZING BLOCKS )
      IF(LBLOCK.LE.0) GOTO 112
      WRITE(LW,520)
  520 FORMAT(/,'INPUT COORDINATES OF NODES CHARACTERIZING BLOCK',
     1 //5X,'NUMBER OF NODES')
      READ(LR,502) NBX
      WRITE(LW,522)
  522 FORMAT(/,'< COORDINATES : XB(I),YB(I) >')
      DO 524 I=1,NBX
      WRITE(LW,526) I
  526 FORMAT('NODE ',I4)
      READ(LR,508) XB(I),YB(I)
  508 FORMAT(2F10.4)
  524 CONTINUE

C     ( READ DATA FOR BLOCKS )

      CALL READBL(LBLOCK,MATPR,MMX,MMY,LLXY,IIJB,PPX,PPY,LLMN,
     1            LR,LW)

      NX=0
      NELX=0
      DO 100 NBLOCK=1,LBLOCK
      MATPRO=MATPR(NBLOCK)
      MX=MMX(NBLOCK)
      MY=MMY(NBLOCK)
      PX=PPX(NBLOCK)
      PY=PPY(NBLOCK)
      LXY=LLXY(NBLOCK)
      DO 116 I=1,LXY
  116 IJB(I)=IIJB(I,NBLOCK)

      CALL ATMESH(NBLOCK,NX,NELX,X,Y,XB,YB,IJK,MPE,NODE,
     1            MATPRO,MX,MY,PX,PY,LXY,IJB,LR,LW)

  100 CONTINUE

C     ( DATA READ MANUALLY I.E. WITHOUT THE GENERATOR )

  112 WRITE(LW,506)
  506 FORMAT(/,'DO YOU NEED TO READ DATA MANUALLY ?  (YES/NO)',
     1  '  DO NOT PUT COMMA AT THE END OF YES/NO.')
      READ(LR,510) LHANDR
  510 FORMAT(A4)
```

```
      IF(LHANDR.NE.IYES) GOTO 102
      CALL HRMESH(NX,NELX,X,Y,IJK,MPE,NODE,LR,LW)
C     ( CONNECTION OF TWO DISJOINT BLOCKS IF NECESSARY )
  102 CALL CONECT(NX,NELX,NODE,X,Y,IJK,JOP1,JOP3,JOP4,LR,LW)
C     ( CONDENSATION OF THE ELEMENTS WHOSE AREA IS ALMOST ZERO )
      CALL CONDEN(NELX,IJK,X,Y,3,LW)
C     ( OPTIMIZATION OF THE NUMBERING )
      WRITE(LW,530)
  530 FORMAT(/,'DO YOU OPTIMIZE THE NUMBERING TO HAVE THE MINIMUM',
     1  ' BAND WIDTH ?   (YES/NO)',/,
     2  'DO NOT PUT COMMA AT THE END OF YES/NO.')
      READ(LR,532) LOPTIM
  532 FORMAT(A4)
      IF(LOPTIM.NE.IYES) GOTO 104
      CALL OPTIMN(NX,NELX,3,IJK,JOP1,JOP2,JOP3,JNEW,JOP4,LR,LW)
C     ( CHANGE THE COORDIINATES )
      DO 108 I=1,NX
      XO(I)=X(I)
  108 YO(I)=Y(I)
      DO 110 I=1,NX
      II=JNEW(I)
      X(II)=XO(I)
  110 Y(II)=YO(I)
C     ( PRINT OUT )
  104 WRITE(LW,602)
  602 FORMAT(/////5X,'********** FINITE ELEMENT MODEL **********',//)
      WRITE(LW,604) NX,NELX
  604 FORMAT(10X,'THE TOTAL NUMBER OF NODAL POINTS =',I5,/10X,
     1 'THE TOTAL NUMBER OF FINITE ELEEMENTS =',I5,////10X,
     2 '< COORDINATES >',/)
      WRITE(LW,606) (I+npnode,X(I),Y(I),I=1,NX)
  606 FORMAT(2(5X,I5,3X,'(',2F10.4,2X,')'))
      WRITE(LW,608)
  608 FORMAT(/////10X,'< ELEMENT CONNECTIVITIES >',/)
      DO 106 I=1,NELX
  106 WRITE(LW,610) I+npelem,MPE(I),(IJK(J,I)+npnode,J=1,NODE)
  610 FORMAT(15X,I5,5X,'<MPE>',I4,5X,'<IJK>',12I5)
C     ( STORE THE DATA )
 1000 continue
      WRITE(LP,800) NX,NELX
```

```
  800 FORMAT(2I5)
      DO 802 I=1,NX
  802 WRITE(LP,804) i+npnode,X(I),Y(I)
  804 FORMAT(I5,2F10.4)
      DO 806 I=1,NELX
  806 WRITE(LP,808) I+npelem,(IJK(J,I)+npnode,J=1,NODE),mpe(i)
  808 FORMAT(16I5)


      STOP
      END
C-----------------------------------------------------------------------
      SUBROUTINE READBL(LBLOCK,MATPR,MMX,MMY,LLXY,IIJB,PPX,PPY,LLMN,
     1                  LR,LW)
C-----------------------------------------------------------------------
C    PURPOSE:
C        READ DATA OF BLOCKS AND CHECK THE DATA

C    ARGUMENT:
C        LBLOCK = TOTAL NUMBER OF BLOCKS
C        MATPR  = GROUP NUMBER OF MATERIALS
C        MMX    = NUMBER OF DIVISIONS IN THE 1-ST DIRECTION
C        MMY    = NUMBER OF DIVISIONS IN THE 2-ND DIRECTION
C        LLXY   = TOTAL NUMBER OF "NODES" IN A BLOCK ( 4,8 OR 12 )
C        IIJB   = BLOCK CONNECTIVITIES
C        PPX    = MESHH GRADIEENT IN THE 1-ST DIRECTION
C        PPY    = MESHH GRADIEENT IN THE 2-ND DIRECTION
C        LLMN   = NODE CONNECTIVITIES

C    INPUT DATA TO BE READ:
C        DO 100 I=1,LBLOCK
C        1. MATPR(N)                          :   I5
C        2. MMX(N),MMY(N)                     :   2I5
C        3. LLXY(N)                           :   I5
C        4. IIJB(I,N),I=1,LLXY(N)             :   I5
C        5. PPX(N),PPY(N)                     :   2F10.4
C-----------------------------------------------------------------------

      DIMENSION MATPR(1),MMX(1),MMY(1),LLXY(1),PPX(1),PPY(1),
     1          IIJB(12,1),LLMN(4,1)

C     ( READ INPUT DATA )

      DO 100 N=1,LBLOCK

      WRITE(LW,500) N
  500 FORMAT(////,'----- INPUT DATA FOR THE BLOCK',I4,' -----',/)
      WRITE(LW,502)
  502 FORMAT(/,'INPUT THE GROUP NUMBER(1-99) OF MATERIALS')
      READ(LR,504) MATPR(N)
  504 FORMAT(I5)
      WRITE(LW,506)
  506 FORMAT(/,'INPUT THE NUMBER OF DIVISION IN THE 1-ST AND 2-ND',
     1  'DIRECTIONS : MMX,MMY')
```

```
      READ(LR,508) MMX(N),MMY(N)
  508 FORMAT(2I5)
      WRITE(LW,510)
  510 FORMAT(/,'INPUT THE NUMBER OF NODES CHARACTERIZING THE BLOCK',
     1  ' : 4, 8, OR 12')
      READ(LR,512) LLXY(N)
  512 FORMAT(I5)
      WRITE(LW,514)
  514 FORMAT(/,'INPUT THE BLOCK CONNECTIVITIY')
      IMAX=LLXY(N)
      DO 112 I=1,IMAX
      WRITE(LW,516) I
  516 FORMAT('IJB(',I3,')')
      READ(LR,518) IIJB(I,N)
  518 FORMAT(I5)
  112 CONTINUE
      WRITE(LW,520)
  520 FORMAT(/,'INPUT REAL PARAMETERS OF MESH GRADIENT IN THE 1-ST',
     1  ' AND 2-ND DIRECTION.',/10X,
     2  'IF PPX = 2.  THEN THE GRID GRADUALLY BECOMES LARGE.',/10X,
     3  'IF PPX = 1.  THEN THE GRID IS UNIFORM.',/10X,
     4  'IF PPX = 0.5 THEN THE GRID GRADUALLY BECOMES SMALL.',/)
      READ(LR,522) PPX(N),PPY(N)
  522 FORMAT(2F10.4)

  100 CONTINUE

C     ( CHECK THE DATA )

      DO 114 N=1,LBLOCK
      DO 114 J=1,4
  114 LLMN(J,N)=0

      DO 116 N=1,LBLOCK
      DO 116 I=1,4
      I1=IIJB(I,N)
      I3=I+1
      IF(I.EQ.4) I3=1
      I2=IIJB(I3,N)
      NN=0
  118 NN=NN+1
      IF(NN.GT.LBLOCK) GOTO 116
      DO 122 J=1,4
      J1=IIJB(J,NN)
      J3=J+1
      IF(J.EQ.4) J3=1
      J2=IIJB(J3,NN)
      IF(I1.EQ.J2.AND.I2.EQ.J1) LLMN(I,N)=10*NN+J
  122 CONTINUE
      GOTO 118
  116 CONTINUE

      LBLOC1=LBLOCK-1
      DO 180 NES=1,LBLOC1
      NMAX=LBLOCK-NES+1
```

```
      DO 130 N=1,NMAX
      DO 130 I=1,4
      M=LLMN(I,N)/10
      IF(M.GE.N) GOTO 130
      L=LLMN(I,N)-10*M
      IF(I.NE.1.OR.L.NE.1) GOTO 132
      PPX(M)=1./PPX(N)
      MMX(M)=MMX(N)
      GOTO 130
  132 IF(I.NE.1.OR.L.NE.2) GOTO 134
      PPY(M)=1./PPX(N)
      MMY(M)=MMX(N)
      GOTO 130
  134 IF(I.NE.1.OR.L.NE.3) GOTO 136
      PPX(M)=PPX(N)
      MMX(M)=MMX(N)
      GOTO 130
  136 IF(I.NE.1.OR.L.NE.4) GOTO 138
      PPY(M)=PPX(N)
      MMY(M)=MMX(N)
      GOTO 130
  138 IF(I.NE.2.OR.L.NE.1) GOTO 140
      PPX(M)=1./PPY(N)
      MMX(M)=MMY(N)
      GOTO 130
  140 IF(I.NE.2.OR.L.NE.2) GOTO 142
      PPY(M)=1./PPY(N)
      MMY(M)=MMY(N)
      GOTO 130
  142 IF(I.NE.2.OR.L.NE.3) GOTO 144
      PPX(M)=PPY(N)
      MMX(M)=MMY(N)
      GOTO 130
  144 IF(I.NE.2.OR.L.NE.4) GOTO 146
      PPY(M)=PPY(N)
      MMY(M)=MMY(N)
      GOTO 130
  146 IF(I.NE.3.OR.L.NE.1) GOTO 148
      PPX(M)=PPX(N)
      MMX(M)=MMX(N)
      GOTO 130
  148 IF(I.NE.3.OR.L.NE.2) GOTO 150
      PPY(M)=PPX(N)
      MMY(M)=MMX(N)
      GOTO 130
  150 IF(I.NE.3.OR.L.NE.3) GOTO 152
      PPX(M)=1./PPX(N)
      MMX(M)=MMX(N)
      GOTO 130
  152 IF(I.NE.3.OR.L.NE.4) GOTO 154
      PPY(M)=1./PPX(N)
      MMY(M)=MMX(N)
      GOTO 130
  154 IF(I.NE.4.OR.L.NE.1) GOTO 156
      PPX(M)=PPY(N)
```

```
      MMX(M)=MMY(N)
      GOTO 130
  156 IF(I.NE.4.OR.L.NE.2) GOTO 158
      PPY(M)=PPY(N)
      MMY(M)=MMY(N)
      GOTO 130
  158 IF(I.NE.4.OR.L.NE.3) GOTO 160
      PPX(M)=1./PPY(N)
      MMX(M)=MMY(N)
      GOTO 130
  160 PPY(M)=1./PPY(N)
      MMY(M)=MMY(N)
  130 CONTINUE
  180 CONTINUE

C     ( OUTPUT )

      WRITE(LW,606)
  606 FORMAT(/////10X,'---------- FINAL INPUT DATA FOR BLOCKS',
     1  ' ----------')
      DO 162 N=1,LBLOCK
      WRITE(LW,600) N
  600 FORMAT(//10X,'----- BLOCK : ',I3,' -----',/)
      WRITE(LW,602) MMX(N),MMY(N),PPX(N),PPY(N)
  602 FORMAT(/,10X,'MX =',I4,5X,'MY =',I4,/10X,'PX =',F10.4,5X
     1 ,'PY =',F10.4,/)
      IMAX=LLXY(N)
      WRITE(LW,604) (IIJB(I,N),I=1,IMAX)
  604 FORMAT(10X,'<IJB>',2X,15I5)
  162 CONTINUE

      RETURN
      END

C-----------------------------------------------------------------------
      SUBROUTINE ATMESH(NBLOCK,NX,NELX,X,Y,XC,YC,IJK,MPE,NODE,
     1                  MATPRO,MX,MY,PX,PY,LXY,IJB,LR,LW)
C-----------------------------------------------------------------------

C    PURPOSE:
C        CREATE THE FINITE ELEMENT MODEL USING THREE-NODE TRIANGULAR
C        ELEMENTS.   THE MESH IS ARRANGED AS THE UNION-JACK PATTERN.
C        THAT IS,  A QUADRILATERAL ELEMENT CONSISTS OF
C        FOUR THREE-NODE-TRIANGULAR ELEMENTS.

C    ARGUMENTS:
C        NBLOCK = BLOCK NUMBER
C        NX     = TOTAL NUMBER OF NODAL POINTS
C        NELX   = TOTAL NUMBER OF FINITE ELEMENTS
C        X,Y    = COORDINATES OF NODAL POINTS
C        XC,YC  = COORDINATES OF "NODES" OF THE BLOCKS
C        IJK    = ELEMENT CONNECTIVITIES
C        MPE    = GROUP NUMBER OF THE MATERIALS FOR EACH ELEMENT
C        NODE   = TOTAL NUMBER OF NODAL POINTS IN AN ELEMENT
C        MATPRO = GROUP NUMBER OF MATERIALS FOR THE BLOCK
```

```
C       MX      = NUMBER OF DIVISION IN THE 1-ST DIRECTION
C       MY      = NUMBER OF DIVISION IN THE 2-ND DIRECTION
C       PX      = PARAMETER OF THE DIVISION FOR THE 1-ST DIRECTION
C       PY      = PARAMETER OF THE DIVISION FOR THE 2-ND DIRECTION
C        (EXAMPLE)    IF  PX=1,    THEN MESH IS EQUALLY DIVIDED
C                     IF  PX=2,    THEN MESH IS GRADUALLY LARGER
C                     IF  PX=0.5,  THEN MESH IS GRADUALLY SMALLER
C       LXY     = TOTAL NUMBER OF "NODES" THAT DEFINES THE BLOCK
C                 4, 8, OR 12 IN THIS PROGRAM
C       IJB     = BLOCK CONNECTIVITY
C   LOCAL VARIALES:
C       NODES  = NUMBER OF THE FIRST NODE OF THE BLOCK
C       NELMS  = NUMBER OF THE FIRST ELEMENT OF THE BLOCK
C       XB,YB  = COORDINATES OF THE "NODES" IN THE BLOCK

C   DIMENSIONING:
C       XB,YB,SH   MUST BE DIMENSIONED FOR THE CASE THAT MORE
C       SOPHISTICATED CURVED BLOCKS ARE ASSUMED THAN 12-NODE
C       SERENDIPITY BLOCK.

C   NOTES:
C       THE NUMBERING SYSTEM FOR EACH ELEMENT IS

C             L2 *---------------* L4
C                I       -N3-    I
C                I               I
C                I -N4-   *  -N2- I
C                I       L5      I
C                I       -N1-    I
C             L1 *---------------* L3

C-----------------------------------------------------------------

      DIMENSION X(1),Y(1),IJK(NODE,1),MPE(1),XC(1),YC(1)
      DIMENSION XB(12),YB(12),SH(12),IJB(12)

C     ( READ INPUT DATA FOR A QUADRILATERAL BLOCK )

      WRITE(LW,598) NBLOCK
  598 FORMAT(/////,'---------- BLOCK ',I3,' ----------',//)

C     ( SET CORRDINATES OF A BLOCK )

      DO 514 I=1,LXY
      IJBI=IJB(I)
      XB(I)=XC(IJBI)
  514 YB(I)=YC(IJBI)

      MX1=MX+1
      MY1=MY+1

C     ( SET THE LAST NODE AND ELEMENT NUMBER )

      NODES=NX+1
```

```
      NELMS=NELX+1
      NX=NODES-1+MX1*MY1+MX*MY
      NELX=NELMS-1+4*MX*MY
      NEQ=2*NX

      DO 100 IX=1,MX
      DO 100 IY=1,MY

C     ( CONSTRUCT THE ELEMENT CONNECTIVITY )

      L1=NODES-1+(IX-1)*MY1+(IX-1)*MY+IY
      L2=L1+1
      L3=L1+MY1+MY
      L4=L3+1
      L5=L1+MY1
      N1=NELMS+4*((IX-1)*MY+(IY-1))
      N2=N1+1
      N3=N1+2
      N4=N1+3

      IJK(1,N1)=L1
      IJK(2,N1)=L3
      IJK(3,N1)=L5
      IJK(1,N2)=L3
      IJK(2,N2)=L4
      IJK(3,N2)=L5
      IJK(1,N3)=L4
      IJK(2,N3)=L2
      IJK(3,N3)=L5
      IJK(1,N4)=L2
      IJK(2,N4)=L1
      IJK(3,N4)=L5

C     ( SPECIFY THE NUMBER OF MATERIAL GROUP )

      MPE(N1)=MATPRO
      MPE(N2)=MATPRO
      MPE(N3)=MATPRO
      MPE(N4)=MATPRO

C     ( COORDINATES )

      XL1=FUN1(IX,MX,PX)
      XL2=XL1
      XL3=FUN1(IX+1,MX,PX)
      XL4=XL3
      YL1=FUN1(IY,MY,PY)
      YL2=FUN1(IY+1,MY,PY)
      YL3=YL1
      YL4=YL2

      CALL SEREND(SH,XL1,YL1,LXY)

      X(L1)=DOT(XB,SH,LXY)
      Y(L1)=DOT(YB,SH,LXY)
```

```
      CALL SEREND(SH,XL2,YL2,LXY)

      X(L2)=DOT(XB,SH,LXY)
      Y(L2)=DOT(YB,SH,LXY)

      CALL SEREND(SH,XL3,YL3,LXY)

      X(L3)=DOT(XB,SH,LXY)
      Y(L3)=DOT(YB,SH,LXY)

      CALL SEREND(SH,XL4,YL4,LXY)

      X(L4)=DOT(XB,SH,LXY)
      Y(L4)=DOT(YB,SH,LXY)
      X(L5)=0.25*(X(L1)+X(L2)+X(L3)+X(L4))
      Y(L5)=0.25*(Y(L1)+Y(L2)+Y(L3)+Y(L4))

  100 CONTINUE

C     ( PRINT OUT THE MESH DATA FOR THE PRESENT BLOCK )

      WRITE(LW,606)
  606 FORMAT(////10X,'< ELEMENT CONNECTIVITIES >',/)
      WRITE(LW,608) (I,MPE(I),(IJK(J,I),J=1,3),I=NELMS,NELX)
  608 FORMAT(5X,I5,3X,I2,4X,'<IJK>',3I5,6X,I5,3X,I2,4X,'<IJK>',3I5)
      WRITE(LW,610)
  610 FORMAT(///15X,'< COORDINATES >',/)
      WRITE(LW,612) (I,X(I),Y(I),I=NODES,NX)
  612 FORMAT(5X,I5,2X,'<',2F10.4,'>',5X,I5,2X,'<',2F10.4,
     1 1X,'>')

      RETURN
      END

C-----------------------------------------------------------------------
      FUNCTION DOT(A,B,N)
C-----------------------------------------------------------------------
C   PURPOSE:
C      TAKE DOT PRODUCT OF TWO VECTORS "A" AND "B"
C-----------------------------------------------------------------------

      DIMENSION A(1),B(1)

      DOT=0.
      DO 100 I=1,N
  100 DOT=DOT+A(I)*B(I)

      RETURN
      END

C-----------------------------------------------------------------------
      FUNCTION FUN1(I,M,P)
C-----------------------------------------------------------------------
C   PURPOSE:
```

```
C       DEFINE THE SIZE OF THE ELEMENT.

C   ARGUMENTS:
C       I   = I-TH ELEMENT
C       M   = TOTAL NUMBER OF ELEMENTS IN EACH DIRECTION
C       P   = 2 ,   IF THE SIZE GRADUALLY BECOMES LARGER
C           = 1 ,   IF THE SIZE IS UNIFORM
C           =0.5,   IF THE SIZE GRADUALLY BECOMES SMALLER
C-------------------------------------------------------------
      RI=I
      RM=M

      IF(P.LE.1.) GOTO 100
      FUN1=-1.+2.*(RI-1.)*RI/(RM*(RM+1.))
      RETURN

  100 IF(P.LT.1) GOTO 102
      FUN1=-1+2.*(RI-1.)/RM
      RETURN

  102 FUN1=-1.+2.*(RI-1.)*(2.*RM-RI+2.)/(RM*(RM+1.))

      RETURN
      END

C-------------------------------------------------------------
      SUBROUTINE SEREND(SH,S,T,N)
C-------------------------------------------------------------
C   PURPOSE:
C       CONSTRUCT THE SHAPE FUNCTIONS OF A SERENDIPITY ELEMENT
C       FOR THE MESH GENERATOR   ( N=4, 8 OR 12 )

C   ARGUMENTS:
C       SH  = SHAPE FUNCTIONS
C       S   = FIRST LOCAL COORDINATE  ( -1, 1 )
C       T   = SECOND LOCAL COORDINATE ( -1, 1 )
C       N   = TOTAL NUMBER OF "NODES"
C           = 4 FOR THE QUADRILATERAL BLOCK
C           = 8 OR 12 FOR CURVED QUADRILATERAL BLOCKS

C   NUMBERING SYSTEM:

C             4-------7-------3         4---10----9----3
C             I               I         I               I
C             I               I        11               8
C             8               6         I               I
C             I               I        12               7
C             I               I         I               I
C             1-------5-------2         1----5----6----2

C-------------------------------------------------------------

      DIMENSION SH(12)
```

```
      KKK=N/4
      GOTO (100,200,300), KKK

C     ( FOR 4-NODE CASE )

  100 SH(1)=0.25*(1.-S)*(1.-T)
      SH(2)=0.25*(1.+S)*(1.-T)
      SH(3)=0.25*(1.+S)*(1.+T)
      SH(4)=0.25*(1.-S)*(1.+T)
      RETURN

C     ( FOR 8-NODE CASE )

  200 SS=S**2
      TT=T**2
      ST=S*T
      SST=SS*T
      STT=S*TT
      SH(1)=(-1.+ST+SS+TT-SST-STT)/4.
      SH(2)=(-1.-ST+SS+TT-SST+STT)/4.
      SH(3)=(-1.+ST+SS+TT+SST+STT)/4.
      SH(4)=(-1.-ST+SS+TT+SST-STT)/4.
      SH(5)=(1.-T-SS+SST)/2.
      SH(6)=(1.+S-TT-STT)/2.
      SH(7)=(1.+T-SS-SST)/2.
      SH(8)=(1.-S-TT+STT)/2.
      RETURN

C     ( FOR 12-NODE CASE )

  300 S1=1.-S
      S2=1.+S
      T1=1.-T
      T2=1.+T
      ST=-10.+9.*(S*S+T*T)
      SH(1)=S1*T1*ST/32.
      SH(2)=S2*T1*ST/32.
      SH(3)=S2*T2*ST/32.
      SH(4)=S1*T2*ST/32.
      SH(5)=9.*(1.-3.*S)*S1*S2*T1/32.
      SH(6)=9.*(1.+3.*S)*S1*S2*T1/32.
      SH(7)=9.*S2*(1.-3.*T)*T1*T2/32.
      SH(8)=9.*S2*(1.+3.*T)*T1*T2/32.
      SH(9)=9.*(1.+3.*S)*S1*S2*T2/32.
      SH(10)=9.*(1.-3.*S)*S1*S2*T2/32.
      SH(11)=9.*S1*(1.+3.*T)*T1*T2/32.
      SH(12)=9.*S1*(1.-3.*T)*T1*T2/32.

      RETURN
      END

C-----------------------------------------------------------------------
      SUBROUTINE HRMESH(NX,NELX,X,Y,IJK,MPE,NODE,LR,LW)
C-----------------------------------------------------------------------
C   PURPOSE:
```

```
C       READ THE MESH DATA ( ELEMENT CONNECTIVITIES AND COORDINATES )
C       WITHOUT USING THE AUTOMETIC MESH GENERATOR.

C   ARGUMENTS:
C       NX    = TOTAL NUMBER OF NODAL POINTS
C       NELX  = TOTAL NUMBER OF FINITE ELEMENTS
C       X,Y   = COORDINATES OF NODAL POINTS
C       IJK   = ELEMENT CONNECTIVITIES
C       MPE   = GROUP NUMBER OF THE MATERIALS FOR EACH ELEMENT
C       NODE  = TOTAL NUMBER OF NODDAL POINTS IN AN ELEMENT

C   LOCAL VARIABLES:
C       NODES = NUMBER OF THE FIRST NODAL POINT ADDED
C       NELMS = NUMBER OF THE FIRST ELEMENT ADDED
C       NODADD= TOTAL NUMBER OF ADDITIONAL NODAL POINTS
C       NELADD= TOTAL NUMBER OF ADDITIONAL ELEMENTS

C   INPUT DATA TO BE READ:
C       1. NODADD                          :   I5
C       2. NELADD                          :   I5
C       3. I,X(I),Y(I),I=1,NODADD          :   I5,2F10.4
C       4. I,MPE(I),(IJK(J,I),J=1,3)
C          K=1,NELADD                      :   5I5
C-------------------------------------------------------------------

      DIMENSION X(1),Y(1),IJK(NODE,1),MPE(1)

      WRITE(LW,500)
500 FORMAT(////,'----- (MANUAL) INPUT DATA -----')
    WRITE(LW,502)
502 FORMAT(/,'INPUT THE TOTAL NUMBER OF NODES INPUT.')
    READ(LR,504) NODADD
504 FORMAT(16I5)
    WRITE(LW,506)
506 FORMAT(/,'INPUT THE TOTAL NUMBER OF ELEMENTS INPUT')
    READ(LR,504) NELADD

    IF(NODADD.LE.0) GOTO 110
    WRITE(LW,508)
508 FORMAT(//,'INPUT COORDINATES OF NODES MODIFIED/ADDED',/)
    DO 100 I=1,NODADD
    WRITE(LW,510)
510 FORMAT('NODE NUMBER,X-COORDINATE,Y-COORDINATE')
    READ(LR,512) II,X(II),Y(II)
512 FORMAT(I5,2F10.4)
    IF(II.GT.NX) NX=II
100 CONTINUE

110 IF(NELADD.LE.0) GOTO 112
    WRITE(LW,514)
514 FORMAT(//,'INPUT THE GROUP NUMBER AND ELEMENT CONNECTIVITY.',/)
    DO 102 I=1,NELADD
    WRITE(LW,516)
516 FORMAT('ELEMENT NUMBER,GROUP NUMBER,<IJK>')
    READ(LR,518) II,MPE(II),(IJK(J,II),J=1,NODE)
```

```
  518 FORMAT(16I5)
      IF(II.GT.NELX) NELX=II
  102 CONTINUE
  112 CONTINUE

      RETURN
      END

C-----------------------------------------------------------------------
      SUBROUTINE CONECT(NX,NELX,NODE,X,Y,IJK,JOP,IBLO1,IBLO2,LR,LW)
C-----------------------------------------------------------------------
C    PURPOSE:
C        CONNECT TWO-DISJOINT BLOCKS BY RENUMBERING THE NODE NUMBERS.

C    ARGUMENTS:
C        NX    = TOTAL NUMBER OF NODAL POINTS
C        NELX  = TOTAL NUMBER OF ELEMENTS
C        NODE  = TOTAL NUMBER OF NODAL POINTS IN AN ELEMENT
C        IJK   = ELEMENT CONNECTIVITIES
C        JOP   = ARRAY TO STORE NODE NUMBERS
C        IBLO1 = NODE NUMBERS OF THE 1-ST BLOCK
C        IBLO2 = NODE NUMBERS OF THE 2-ND BLOCK

C    LOCAL VARIABLES:
C        NCX   = TOTAL NUMBER OF NODAL POINTS CONNECTED
C-----------------------------------------------------------------------

      DIMENSION X(1),Y(1),IJK(NODE,1),JOP(1),IBLO1(1),IBLO2(1)
      DATA TOR/1.E-5/

C     ( FIND THE NODES OVERLAPPED )

      NXO=NX
      NCX=0
      DO 300 I=1,NX
      JOP(I)=I
      XI=X(I)
      YI=Y(I)
      J1=I+1
      DO 302 J=J1,NX
      XIJ=ABS(XI-X(J))
      YIJ=ABS(YI-Y(J))
      IF(XIJ.GT.TOR.OR.YIJ.GT.TOR) GOTO 302
      NCX=NCX+1
      IBLO1(NCX)=I
      IBLO2(NCX)=J
  302 CONTINUE
  300 CONTINUE
      IF(NCX.LE.0) RETURN

      WRITE(LW,600) NCX
  600 FORMAT(/////10X,'----- CONNECTION -----',5X,'NCX=',I5,//)
      WRITE(LW,602) (I,IBLO1(I),IBLO2(I),I=1,NCX)
  602 FORMAT(10X,I5,5X,'(',2I5,' )')
```

```
C      ( RENUMBERING )

       NC=0
  210  NC=NC+1
       I1=IBLO1(NC)
       I2=IBLO2(NC)
       IF(I1-I2) 202,200,204
  202  IBLO2(NC)=I1
       NNODE=I1
       NPIVOT=I2
       GOTO 206
  204  IBLO1(NC)=I2
       NNODE=I2
       NPIVOT=I1
  206  NC1=NC+1
       DO 208 I=NC1,NCX
       IA=IBLO1(I)
       IB=IBLO2(I)
       IF(IA.EQ.NPIVOT) IBLO1(I)=NNODE
       IF(IB.EQ.NPIVOT) IBLO2(I)=NNODE
       IF(IA.GT.NPIVOT) IBLO1(I)=IA-1
       IF(IB.GT.NPIVOT) IBLO2(I)=IB-1
  208  CONTINUE
       DO 120 NEL=1,NELX
       DO 120   I=1,NODE
       IJKI=IJK(I,NEL)
       IF(IJKI.EQ.NPIVOT) IJK(I,NEL)=NNODE
       IF(IJKI.GT.NPIVOT) IJK(I,NEL)=IJKI-1
  120  CONTINUE
       DO 114 I=1,NXO
       JOPI=JOP(I)
       IF(JOPI.EQ.NPIVOT) JOP(I)=NNODE
       IF(JOPI.GT.NPIVOT) JOP(I)=JOPI-1
  114  CONTINUE
       NOD1=NPIVOT+1
       DO 112 I=NOD1,NX
       X(I-1)=X(I)
  112  Y(I-1)=Y(I)
       NX=NX-1

       NC2=NC+2
       IF(NC2.GT.NCX) GOTO 200
       DO 212 N=NC1,NCX
       I1=IBLO1(N)
       I2=IBLO2(N)
       N1=N+1
       NCOUNT=0
       DO 214 M=N1,NCX
       IF(IBLO1(M).NE.I1.OR.IBLO2(M).NE.I2) GOTO 214
       NCOUNT=NCOUNT+1
       DO 216 L=M,NCX
       IBLO1(L)=IBLO1(L+1)
  216  IBLO2(L)=IBLO2(L+1)
  214  CONTINUE
       NCX=NCX-NCOUNT
```

```
  212 CONTINUE
  200 IF(NC.LT.NCX) GOTO 210

C     ( PRINT OUT )

      WRITE(LW,620)
  620 FORMAT(/////,10X,'----- CONDENSED NODE NUMBERS -----',/)
      WRITE(LW,622) (I,JOP(I),I=1,NXO)
  622 FORMAT(5(5X,'(',I5,'  -',I5,' )'))

      RETURN
      END

C-----------------------------------------------------------------------
      SUBROUTINE CONDEN(NELX,IJK,X,Y,NODE,LW)
C-----------------------------------------------------------------------

C   PURPOSE:
C       TAKE OUT THE ELEMENTS IF THEY HAVE NON-POSITIVE DETERMINANT.

C   ARGUMENTS:
C       NELX  = TOTAL NUMBER OF ELEMENTS
C       IJK   = ELEMENT CONNECTIVITIES
C       X,Y   = COORDINATES OF NODAL POINTS
C       NODE  = TOTAL NUMBER OF NODAL POINTS IN AN ELEMENT

C   LOCAL VARIABLES:
C       RMIN  = RADIUS OF CIRCLE IN A TRIANGLE
C       RMAX  = RADIUS OF CIRCLE CIRCUMSCRIBING A TRIANGLE
C       RATIO = RMIN/RMAX
C       TOR1  = TOLERENCE TO ZERO AREA (1.E-4)
C       TOR2  = MINIMUM VALUE OF RATIO ALLOWANCE
C-----------------------------------------------------------------------

      DIMENSION IJK(NODE,1),X(1),Y(1)
      DATA TOR1,TOR2/1.E-4,0.01/

      NEL=0
  100 NEL=NEL+1
      I1=IJK(1,NEL)
      I2=IJK(2,NEL)
      I3=IJK(3,NEL)
      X1=X(I1)
      Y1=Y(I1)
      X2=X(I2)
      Y2=Y(I2)
      X3=X(I3)
      Y3=Y(I3)
      A=SQRT((X1-X2)*(X1-X2)+(Y1-Y2)*(Y1-Y2))
      B=SQRT((X2-X3)*(X2-X3)+(Y2-Y3)*(Y2-Y3))
      C=SQRT((X3-X1)*(X3-X1)+(Y3-Y1)*(Y3-Y1))
      S=0.5*(A+B+C)
      IF(A.LE.TOR1) GOTO 102
      IF(B.LE.TOR1) GOTO 102
      IF(C.LE.TOR1) GOTO 102
```

```
C      ( IF AREA IS POSITIVE )

       AREA=0.5*(X1*(Y2-Y3)+X2*(Y3-Y1)+X3*(Y1-Y2))
       RMIN=AREA/S
       RMAX=0.25*A*B*C/AREA
       RATIO=RMIN/RMAX
       IF(RATIO.LE.TOR2) WRITE(LW,600) NEL,RMIN,RMAX,AREA
  600  FORMAT(//,'***** INADEQUATE TRIANGULAR ELEMENT',5X,'NEL = ',I5,
     1 5X,'RMIN = ',E10.3,5X,'RMAX = ',E10.3,5X,'AREA = ',E10.3)
       GOTO 108

C      ( IF AREA IS CLOSE TO ZERO )

  102  NEL1=NEL+1
       IF(NEL1.GT.NELX) GOTO 106
       DO 104 LEL=NEL1,NELX
       DO 104 IA=1,NODE
  104  IJK(IA,LEL-1)=IJK(IA,LEL)
  106  NELX=NELX-1
       NEL=NEL-1

  108  IF(NEL.LT.NELX) GOTO 100

       RETURN
       END

C-----------------------------------------------------------------------
      SUBROUTINE OPTIMN(NX,NELX,NODE,IJK,JM,MT,JO,JP,NW,LR,LW)
C-----------------------------------------------------------------------
C   PURPOSE:
C       OPTIMIZE THE NUMBERING TO HAVE THE MINIMUM BAND WIDTH
C       OF A FINITE ELEMENT MODEL BY 3-NODE ELEMENTS.

C   ARGUMENTS:
C       NX     = TOTAL NUMBER OF NODAL POINTS
C       NELX   = TOTAL NUMBER OF ELEMENTS
C       NODE   = TOTAL NUMBER OF NODAL POINTS IN AN ELEMENT
C       IJK    = ELEMENT CONNECTIVITIES
C       JM     = STORES THE NUMBER OF ELEMENTS RELATED TO AN ELEMENT
C       MT     = CONTAINS THE ELEMENT NUMBERS RELATED TO A NODAL POINT
C       JO     = CONTAINS THE OLD NUMBERING OF NODAL POINTS
C       JP     = CONTAINS THE NEW NUMBERING OF NODAL POINTS
C       NW     = DUMMY ARRAY
C-----------------------------------------------------------------------
       DIMENSION IJK(NODE,1),JM(1),MT(1),JO(1),JP(1),NW(1)

C      ( RENUMBERING )

       IMAX=8*NX
       DO 132 I=1,IMAX
  132  MT(I)=0
       MB=0
       DO 100 J=1,NX
```

```
  100 JM(J)=0
      DO 102 J=1,NELX
      DO 104 I=1,NODE
      JN=IJK(I,J)
      JS=8*(JN-1)
      DO 106 II=1,NODE
      IF(I.EQ.II) GOTO 106
      JJ=IJK(II,J)
      ME=JM(JN)
      IF(ME.EQ.0) GOTO 108
      DO 110 I3=1,ME
      IF(MT(JS+I3).EQ.JJ) GOTO 106
  110 CONTINUE
  108 JM(JN)=JM(JN)+1
      MT(JS+JM(JN))=JJ
      IF(IABS(JN-JJ).GT.MB) MB=IABS(JN-JJ)
  106 CONTINUE
  104 CONTINUE
  102 CONTINUE

      WRITE(LW,600) MB
  600 FORMAT(/////5X,'---------- OPTIMIZATION OF THE NUMBERING',
     1 ' ----------',//10X,'BAND WIDTH BEFORE THE OPTIMIZATION = ',I5)

      NJ=NX
      MI=MB
      DO 112 IK=1,NX
      DO 114 J=1,NX
      JO(I)=0
  114 NW(J)=0
      MA=0
      I=1
      NW(1)=IK
      JO(IK)=1
      K=1
  122 K4=JM(NW(I))
      IF(K4.LE.0) GOTO 116
      JS=8*(NW(I)-1)
      DO 118 JJ=1,K4
      K5=MT(JS+JJ)
      IF(JO(K5).GT.0) GOTO 118
      K=K+1
      NW(K)=K5
      JO(K5)=K
      ND=IABS(I-K)
      IF(ND.GE.MI) GOTO 112
      IF(MA.LT.ND) MA=ND
  118 CONTINUE
      IF(K.EQ.NJ) GOTO 120
  116 I=I+1
      IF(I.GT.NX) GOTO 120
      GOTO 122
  120 MI=MA
      DO 124 J=1,NX
  124 JP(J)=JO(J)
```

```
  112 CONTINUE

      INDEX=0
      DO 134 I=1,NX
  134 IF(JP(I).LE.0) INDEX=INDEX+1
      IF(INDEX.EQ.0) GOTO 138
      WRITE(LW,612)
  612 FORMAT(///,'***** OPTIMIZER DID NOT WORK FOR THIS CASE *****',//)
      DO 140 I=1,NX
  140 JP(I)=I
      GOTO 136

C     ( OUTPUT OF THE NEW NUMBERING )

  138 WRITE(LW,608)
  608 FORMAT(///10X,'< THE NEW NUMBERING SYSTEM >',/)
      WRITE(LW,610) (I,JP(I),I=1,NX)
  610 FORMAT(3(5X,'(',I5,'  ~',I5,' )'))

C     ( NEW CONNECTIVITIES )

      DO 126 NEL=1,NELX
      DO 126 IA=1,NODE
      IJKIA=IJK(IA,NEL)
  126 IJK(IA,NEL)=JP(IJKIA)

  136 WRITE(LW,602)
  602 FORMAT(///10X,'< OPTIMIZED ELEMENT CONNECTIVITIES >',/)
      WRITE(LW,604) (NEL,(IJK(IA,NEL),IA=1,3),NEL=1,NELX)
  604 FORMAT(10X,I5,3X,'<IJK>',3I5,10X,3X,'<IJK>',3I5)

C     ( NEW BAND WIDTH )

      MB=0
      DO 130 NEL=1,NELX
      DO 130 IA=1,NODE
      IJKIA=IJK(IA,NEL)
      DO 130 JB=1,NODE
      IJKJB=IJK(JB,NEL)
      IAJB=IJKJB-IJKIA+1
      IF(IAJB.GT.MB) MB=IAJB
  130 CONTINUE

      WRITE(LW,606) MB
  606 FORMAT(///10X,'NEW BAND WIDTH = ',I5)

      RETURN
      END
```