기 술 기 사

# C-언어를 이용한 대화형 골조 해석 프로그램의 개발

김 한 수*

## 1. 서 언

건축, 토목 분야에서 컴퓨터를 가장 먼저 적용한 분야가 구조 해석 분야일 것이다. 1960년대 이후부터 많은 구조 해석용 프로그램이 개발되어 왔으며 현재에도 많은 프로그램들이 실무에서 사용되고 있다. 지금까지 개발되어온 대부분의 프로그램들은 해석을 위한 대상 구조물의 자료를 입력화일을 이용하여 표현하는 방법을 사용하고 있다. 이 입력화일의 작성 방법은 각 해석 프로그램마다 상이하여 프로그램의 사용을 위해서는 입력화일 작성을 위한 복잡한 지침서를 참조해야만 하였다. 특히 범용 해석 프로그램은 여러가지의 요소와 해석방법 등을 제공하고 있기 때문에 입력화일 작성의 어려움이 더욱 크다고 하겠다. 일부 프로그램들은 입력화일의 작성 방법에 있어서 자료의 입력 순서에 상관하지 않는다든지, free format 입력 방법 등을 통하여 사용자의 편의를 도모하고 있으나 입력화일 작성의 어려움의 근본적 해결방법은 되지 못하였다.

해석을 위한 구조물의 표현 방법중 현재 가장 편리한 방법은 그래픽을 이용한 방법이라 할 수 있다. 해석 프로그램에서의 그래픽은 주로 해석 결과의 후처리 단계에서 많이 사용되어 왔으며 입력 단계에서의 사용은 입력화일의 확인을 위한 수단으로 사용되어 왔다. 입력화일의 확인을 위하여 그래픽을 사용하는 방법도 입력화일의 정확성을 쉽게 확인시켜 줄 뿐, 입력화일 작성의 경직성은 극복하지 못한다. 입력 그 자체를 그래픽을 이용하는 방법이 가장 자연스러운 표현 방법이 될 것이다. 이러한 그래픽을 이용한 구조물의 표현 방법을 이용하면 절점 번호, 요소 번호 등의 개념이 필요없으며 입력이 곧 확인이 된다. 이러한 입력 방법들을 워드프로세서에 비유한다면 입력화일의 작성은 LaTex과 같은 document formatter에 비유될 수 있고 그래픽을 이용한 방법은 WYSIWYG(What You See Is What You Get) 개념의 워드프로세서에 비유될 수 있다. 즉 그래픽을 이용한 입력 방법은 WYSIWYA(What You See Is What You Analyze) 개념을 적용한 것이라 볼 수 있다. 물론 사용자의 능력에 따라 입력화일을 이용한 방법이 편리할 수도 있다. 그러나 입력화일을 이용한 방법이 편리하게 느껴지려면 사용자가 특정 해석 프로그램을 수많은 시행착오를 거쳐 상당 기간 사용해 왔음에 틀림없다.

다른 분야의 컴퓨터 소프트웨어를 분석해 보면 사용자의 편의성을 강조하는 경향은 쉽게 알 수 있다. 사용지침서를 참조하여 암호같은 명령어를 입력하는 단계에서 점차 메뉴, 다이얼로그 박스, 아이콘 등을 이용한 직관적인 소프트웨어 사용 방법을 채용하는 경향으로 변하였다. 구조해석 프로그램에서도 이러한 경향이 채택되어져야 한다.

현재 그래픽을 가장 효율적으로 구사할 수 있는 컴퓨터 언어는 C-언어라고 할 수 있다. C-언어는 최근 컴퓨터 프로그래밍에 있어서 현재 가장 많이 사용되는 언어이며 어셈블리의 효율성과 파스칼의 표현력을 겸비한 강력한 언어이다. 구조해석 분야에서는 주로 포트란이 사용되어 왔으나 그

* 정회원, 한국과학기술원 토목공학과, 위촉연구원, 공학박사

래픽을 바탕으로 한 해석프로그램의 개발에 있어서는 C-언어가 더욱 적절하다고 판단된다. 필자는 WYSIWYA개념을 적용한 대화식 2차원 골조해석 프로그램인 Visual Frame을 C-언어를 이용하여 개발하였다. Visual Frame의 개발 과정에서 얻은 그래픽을 이용한 구조물의 표현방법과 구조해석 프로그램에의 C-언어 적용에 관한 몇가지 사항을 기술하려 한다.

## 2. 그래픽을 이용한 대화식 골조 해석 프로그램

Visual Frame은 PC의 도스 3.0이상에서 수행되며 VGA와 마우스가 필요하며 메모리의 크기는 불과 350K 정도만이 요구될 뿐이다. 따라서 XT, AT, 386, 486 등 어떠한 PC에서도 운용 가능하다. 대략적인 프로그램의 사양은 표 1에 나타내었다.

Visual Frame에서의 모든 명령은 마우스를 이용한 풀다운 메뉴(Pull-Down Menu)를 통하여 선택된다. 마우스를 화면의 윗부분에 위치시키면 FILE, DRAW, EDIT, DISPLAY, LOADCASE, ANALYSIS, POST 등의 주메뉴가 나타나게 되며 각 메뉴를 선택하면 그에 해당하는 서브메뉴 또는 다아얼로그 박스 등이 나타내게 되어 프로그램을 쉽게 운용할 수 있게 되어 있다. 구조물의 입력은 마우스를 이용하여 직접 화면에 입력시키고 해석은 버턴만 누름으로써 신속하게 이루어지게 된다. 후처리 또한 하나의 프로그램에서 해결된다. 이처럼 하나의 프로그램에서 전처리, 해석, 후처리의 각 단계를 모두 해결할 수 있어 해석의 전 과정을 신속하게 처리할 수가 있게 되어 있다.

표 1. Visual Frame의 사양

| 대 상 구 조 | 평면 골조, 평면 트러스 |
|---|---|
| 보 요 소 | 2,500 |
| 자 유 도 | 4,500 |
| 재 료 종 류 | 10 |
| 단 면 종 류 | 50 |
| 스 프 링 종 류 | 20 |
| 하 중 조 합 | 30 |
| 하 중 그 룹 | 6 |
| 절 점 하 중 종 류 | 30 |
| 보 하 중 종 류 | 30 |

(1) 해석 골조의 형상 입력

Visual Frame에서 보요소는 선분으로 표현된다. DRAW/ELEMENT 메뉴를 선택하면 재료와 단면의 성질을 입력할 수 있고 양 끝점의 연결 관계를 강접 또는 힌지로 선택할 수 있다. 선택이 끝나면 마우스의 커서가 두개의 교차하는 직선으로 바뀌어 보요소를 입력하는 단계임을 알려주며 마우스를 이용하여 선분을 화면에 그리므로써 보요소를 입력할 수 있다. 선분을 그리게 되면 시작점과 끝점은 작은 원으로 절점임을 표시한다. 키보드로도 입력할 수 있는데 시작점과 끝점의 좌표를 입력함으로써 보요소를 입력할 수 있다. 키보드를 사용하면 절대좌표 뿐만 아니라 끝점의 시작점에 대한 상대좌표로도 입력할 수 있다. 전에 그려진 선분들과 현재 그리는 선분과의 교차점은 자동으로 절점으로 인식되며 선분들도 여러개의 보요소로 분리된다. 자료를 입력하는 동안 화면의 크기는 DISPLAY/ZOOM 기능으로 얼마든지 조정할 수 있으며 GRID, SNAP, ORTHO 등의 기능을 이용할 수도 있다. 흡사 Auto CAD에서 골조의 모양을 그리는 방법으로 구조물의 형상을 표현한다. 이러한 과정에서는 절점번호, 요소번호, 요소의 연결관계 등과 같은 개념은 필요하지 않게 된다. 그러한 개념은 Visual Frame의 내부에서만 관리될 뿐이다. 시각적으로 보이는 형태가 바로 해석하려는 구조물이므로 구조물의 형상을 잘못 입력하는 경우는 있을 수가 없다.

(2) 경계조건 및 하중의 입력

경계조건은 DRAW/BOUNDARY 메뉴를 선택하여 X 이동변위, Y 이동변위, Z 회전변위에 대하여 자유와 고정의 경계조건을 결정한 후 원하는 절점에 사각형의 커서를 위치시키고 마우스 버턴을 누름으로써 설정된다. 화면에서는 삼각형과 원형으로 해당 절점의 경계조건이 설정되어 있음을 보여주게 된다. 하중은 절점하중과 보하중의 형태로 줄 수 있는데 다이얼로그 박스에서 그 값을 지정할 수 있다. 보하중의 경우에는 아이콘으로 보하중의 분포 형태를 선택할 수 있도록 하였다(그림 1).

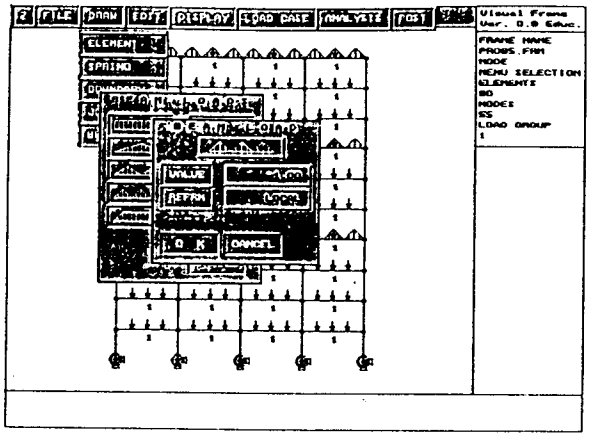

그림 1 Visual Frame에서의 보하중의 입력

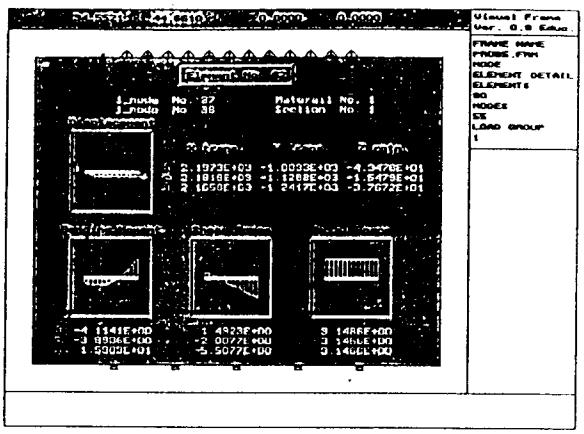

그림 2 Visual Frame에서의 후처리 예

(3) 편집 및 해석

재료상수와 단면계수 등은 요소를 그리기 전에 그 값을 부여할 수도 있고 이미 그려진 요소에 그 값을 부요할 수도 있도록 하였다. 요소, 경계조건, 하중 등은 언제든지 삭제하거나 추가할 수가 있다. 하중은 LOADGROUP과 LOADCASE를 이용하여 자유롭게 조합할 수 있다. 입력 작업이 끝나면 ANALYSIS 메뉴를 선택하여 즉각적으로 해석을 행할 수 있도록 되어 있다. Visual Frame에서는 입력 화일을 읽어들이는 시간이 필요없으므로 전체 해석시간이 상당히 단축되며 연립방정식의 해를 구하는 방법으로 코아내 해법(in-core solver) 보다 빠른 스카이라인 해법을 바탕으로 한 코아외 해법(out-of-core solver)를 개발, 채택하였으므로 어떠한 해석프로그램보다 빠른 해석시간을 보여준다.

(4) 후처리 및 기타 기능

해석이 끝나면 POST 메뉴를 통하여 변형 상태, 축하중, 전단력, 휨모멘트 등의 부재력 그림을 그려볼 수도 있고 원하는 요소를 선택하면 그 요소의 단부와 중앙부의 부재력값을 알아볼 수도 있다. 또한 해석 결과로 나온 출력 화일도 프로그램 내부에서 열람할 수 있도록 되어 있다. 부재력 그림에서는 요소 단부의 연결조건(예를 들어 힌지로 연결된 경우)과 보하중에 의한 부재력 변화를 완벽하게 반영하여 부재력을 도시하여 준다(그림 2).

한편, DISPLAY 메뉴를 통하여 절점번호, 요소번호, 재료번호, 단면번호 등을 확인할 수 있으며 FILE 메뉴에서는 화일의 불러오기, 저장하기, 도스쉘 등의 기능을 이용할 수 있으며 화면의 프린트를 위하여 원하는 화면의 상태를 PCX화일의 형태로 저장할 수 있도록 되어 있다.

일반 사용자를 통한 Visual Frame의 베타 테스트를 실시한 결과 그 사용법을 대부분 스스로 알 수 있어 그래픽을 바탕으로 한 대화형 해석프로그램의 사용자 편의성을 확인할 수 있었다.

## 3. C-언어를 이용한 해석프로그램의 개발

Visual Frame은 대부분 C-언어로 작성되었고 화면의 제어을 위하여 약간의 어셈블리어가 사용되었다. 이같이 C-언어를 이용하여 구조해석 프로그램을 개발할 경우, 포트란을 사용하는 경우와 비교하여 보면 다음과 같은 장점이 있음을 알았다.

(1) 구조체와 사용자 정의 자료 형식의 이용

포트란에서는 정수와 실수와 같은 기본 자료형식이외에 ARRAY 형식을 이용할 수 있는 반면 C-언어에서는 ARRAY 이외에도 구조체를 곤간

으로 한 사용자 정의 자료 형식을 이용할 수 있다. 예를 들어 절점의 경우에는 X좌표값과 Y좌표값이 필요하게 되는데 이를 구조체로 정의하여 사용한다든지, 보요소에 관련된 절점번호, 재료번호, 단면번호, 보하중 번호 등을 하나의 구조체로 정의하여 사용하게 되면 그 자료 관리는 상당히 편리해지며 프로그램의 작성시 각 자료의 크기를 잘못 정의하는 등의 버그를 근본적으로 없앨 수 있게 된다(그림 3).

```
typedef struct {
        char delete ; /* delete flag : YES, NO */
        double x ;  /* x coordinate */
        double y ;  /* y coordinate */
        int     bc[3] ;     /* boundary condition */
        int jload[NUMLLOADGROUP] ;
                /* jointload index for Loadgroup 1···6 */
        int     elastic ;   /* # of elastic support type */
        }NODE ;
typedef struct {
        char    kind ;  /* kind of element    'C' : column */
                        /*                    'B' : beam   */
                        /*                    'I' : inclined */
        char    delete ;  /* delete flag : YES, NO */
        int     i_node ;/* # of i node */
        int     i_node ;/* # of j node */
        int     bload[NUMLLOADGROUP] ;
                /* beamload index for Loadgroup 1···6 */
        int     mat_type ;   /* # of material type */
        int     sec_type ;   /* # of section type */
        int     i_connect ;  /* i node release FIXED or
                                HINGE */
        int     j_connect ;  /* j node release FIXED or
                                HINGE */
        }ELEMENT ;
```

그림 3 절점과 요소의 자료형 정의

(2) 포인터의 사용

어레이을 사용하는 것과 비교하여 포인터를 사용할 때 얻을 수 있는 잇점은 계산 시간이 빨라지는 점도 있지만 그 보다는 동적 메모리 활당을 우선 들 수 있다. 포트란을 이용한 구조 해석 프로그램에서도 동적활당이라는 개념이 있으나 이는 우선 큰 어레이를 정의한 후 이를 실행중에 여러 목적으로 나누어 사용하는 프로그램 내부에서의 동적활당 개념이다. C-언어에서의 동적활당은 프로그램이 실행중에 운영체계로부터 메모리를 활당받는 개념이어서 진정한 의미에서의 동적활당이 이루어 지게 된다. 예를 들어 최대 1,000개의 요소를 해석할 수 있는 프로그램이 작성되었다면 메모리가 줄어든 상태(예를 들어 사용하는 도스 버젼이 다르다던지, 램상주 프로그램이 실행중이라던지, 다른 프로그램에서의 도스쉘 상태)에서는 포트란의 동적활당으로는 프로그램의 실행 자체가 될 수 없지만 C-언어에서의 동적활당을 이용하면 1,000개의 요소는 해석할 수 없지만 남아있는 메모리로 가능한 수의 요소는 해석할 수 있게 된다. 이러한 동적메모리 활당을 이용하면 필요할 때 메모리를 활당받고 필요하지 않는 메모리는 해제할 수 있어 메모리를 상당히 효율적으로 관리할 수 있게 된다.

(3) 시스템 자원의 적극적 활용

포트란은 기본적으로 과학계산용으로 개발된 언어이어서 키보드, 그래픽, 마우스와 같은 현대의 컴퓨터 자원을 이용하기가 쉽지 않게 되어 있다. PC에서 포트란으로 화면의 특정위치에 지정된 색깔로 글자를 출력할 수 있는 프로그램을 작성해 보라. 특수한 라이브러리를 사용한다던지, ANSI의 이스케이프 시퀀스를 이용하지 않으면 불가능할 것이다. 그래픽이 가능하다는 포트란 컴파일러도 알고보면 그래픽 라이브러리는 C-언어에서 채용한 것임을 발견할 수 있다. 계산만 하는 프로그램의 작성의 경우에는 포트란이 우수하다고 할 수도 있지만 사용자 편의성을 고려한 프로그램을 작성하려고 하면 한계에 부닥치게 된다. C-언어는 어셈블리를 대체할 수 있는 저급 수준을 지원하므로 그래픽, 마우스, 키보드 등과 같은 시스템 자원을 마음대로 활용할 수 있게 해주어 사용자 편의성이 극대화된 프로그램을 개발할 수 있도록 되어 있다. 그리고 프로그램의 개발환경을 비교해 보아도 C-언어의 우월성을 알 수 있다. 통합된 소스레벨 디버그, 온라인 도움말 등의 C-언어의 개발환경을 적극적으로 활용하면 프로그램의 개발시간을 상당히 단축시킬 수 있다.

C-언어를 이용하여 구조 해석 프로그램을 개발할 경우, 단점도 있다. 가장 큰 단점은 기존의

코드를 이용하기가 곤란하다는 점이다. 전통적으로 구조 해석 프로그램은 주로 포트란으로 개발되어 있어 이전에 작성된 코드를 그대로 이용할 수가 없고 C-언어로 변역하는 단계를 거쳐야만 한다. 두번째의 단점은 구조해석 프로그램의 개발을 담당하는 구조 기술자들이 대체로 C-언어에 익숙하지 못하다는 점이며 C-언어가 포토란에 비하여 배우기가 어렵다는 점이다. 그러나 사용자 편의성을 고료한 프로그램을 작성하려면 이러한 어려움은 극복되어져야 한다. 프로그래머는 소수이지만 사용자는 다수이다.

### 4. C-언어로 구현된 Visual Frame의 특징

위에서는 C-언어를 이용한 구조해석 프로그램의 개발에 관련된 일반적인 사항을 논하였다. 이번에는 Visual Frame에는 C-언어를 이용하여 구현된 몇가지 특징을 논하고자 한다.

(1) 코아내 해법보다 빠른 코아외 해법

일반적으로 코아내 해법(in-core equation solver)은 램(RAM) 메모리만을 사용하고 코아외 해법(out-of-core equation solver)에서는 전체 강성행렬을 디스크에 저장하고 램을 버퍼로 사용하므로 램과 디스크의 속도 차이때문에 코아외 해법이 느리게 마련이다. 그러나 Visual Frame에서는 코아내 해법보다도 빠른 코아외 해법이 구현되어 있다. 이 신기한 결과의 비밀은 Visual Frame의 코아외 해법에서의 버퍼의 크기를 64K로 제한한 것이다.

우리가 사용하고 있는 PC의 대부분은 인텔의 8086계열의 마이크로 프로세서를 사용하고 있는데, 이들 미크로프로세서는 리얼모드에서 메모리 주소지정을 위하여 세그먼트 : 오프셋 형태로 하기때문에 단일 자료의 크기가 64K를 넘을 경우 huge 포인터라는 것을 이용하여 메모리를 엑서스해야만 한다. 자료의 크기가 64K를 넘을 경우 오프셋의 변경만으로는 정확한 주소지정이 불가능하고 또한 특정 메모리 주소가 여러 형태의 세그먼트 : 오프셋으로 지정될 수 있으므로 정확한 주소지정을 위하여 세그먼트 : 오프셋의 형태를 매번 정규화(normalization)해야 하는 오버헤드가 있게 된다. 따라서 연립방정식의 해법에서의 huge 포인터의 사용은 그 해석 시간을 상당히 지연시키는 요인이 된다. 반면에 버퍼의 크기를 64K로 제한하면 huge 포인터 대신 far 포인터를 사용할 수 있게 되어 주소지정의 정규화에 필요한 오버헤드를 제거할 수 있게 된다.

또한 디스크의 이용에서는 하나의 직접 접근 이진 화일(direct access binary file)을 사용하여 솔버의 최외각 루프에서만 화일 I/O의 필요성이 검토되고 행성행렬 조합, 소거, 역대입 등의 과정을 고려하여 최소한의 화일 접근을 행함으로써 화일 I/O의 횟수를 최소화시킨 알고리듬을 개발하였다. 결론적으로 코아외 해법의 화일 사용으로 인한 수행속도 감소 요인이 huge 포인터 사용으로 인한 수행속도 감소 요인보다 그 효과가 적어 코아내 해법보다 빠른 코아외 해법을 구현할 수 있었다.

(2) 효율적 에러처리

Visual Frame과 같은 대화형 프로그램에서는 에러 처리에 상당한 노력을 기울여야 한다. 대화형 해석 프로그램에서는 경계조건의 오류, 재료, 단면 계수의 잘못된 입력 등으로 인한 해석상의 에러뿐만 아니라 해석 결과가 나오기 전에 POST 메뉴를 선택한다던지 하는 프로그램 운영상의 에러 처리에도 대응할 수 있어야 한다. 대화형 프로그램은 배치형(batch type) 프로그램과 달리 에러가 발생한 경우에도 프로그램의 수행이 중단되어서는 안되며 사용자에게 에러의 발생을 알리고 그 원인을 제거할 수 있는 방법을 제공해야만 한다. 프로그램에서 발행될 수 있는 다양한 에러 처리를 위해서는 일관된 에러 처리 루틴의 작성이 필수적이다. 그러나 에러의 발생은 여러 단계로 분기된 서브루틴에서도 발생될 수 있고 또한 에러 처리 후에는 에러 발생 이전의 상태로 되돌아 와야 하므로 특별한 에러처리 방법이 강구되어야 한다. Visual Frame에서는 C-언어의 setjmp함수와 longjmp함수를 이용하여 에러를 처리하였다. longjmp함수는 서브루틴 내부에서만 점프할 수 있는 goto함수와 달리 여러단계로 분기된 서브루

틴에서 setjmp함수로 지정된 전체 프로그램중의 특정 위치로 점프할 수 있도록 해주눈 함수로서 에러처리 루틴의 작성에 강력한 기능을 발휘한다. longjmp함수를 이용하면 여러단계로 분기된 서브 루틴의 경우에도 스택의 복구, 레지스터의 복구 등을 자동으로 해주어 에러 처리 루틴으로 되돌아 가기 위한 여러번의 return이 필요 없게 된다.

## 5. 결 언

이상에서 C-언어를 사용한 대화형 2차원 골조 해석 프로그램의 개발과정에서 얻은 몇가지 사항을 기술하였다. WYSIWYA 개념이 구조해석 프로그램에 있어서 편리한 사용자 인터페이스 방법의 하나가 될 수 있음을 보였고, 개발에 사용한 C-언어는 다양한 자료 형태, 포인터의 사용, 그래픽을 비롯한 시스템 자원의 적극적 활용 등의 면에 있어서 기존의 포트란에 비하여 그 기능이 우수하다고 판단된다.

다수의 사용자가 이용할 목적으로 개발되는 구조 해석 프로그램은 사용자 편의성이 우선적으로 고려되어야 하며 기존의 입력화일, 해석, 출력화일의 단순한 사용자 인터페이스에서 벗어나 구조해석이라는 특수한 분야에 맞는 새로운 사용자 인터페이스 방법들이 많이 개발되어져야 한다.