〈기술논문〉

OOP 개념에 기초한 유동해석용 후처리 프로그램 개발

명 현 국† · 안 종 기*

(2007년 7월 25일 접수, 2007년 11월 8일 심사완료)

Development of a Post-Processing Program for Flow Analysis Based on the Object-Oriented Programming Concept

Hyon Kook Myong and Jong Ki Ahn

Key Words: Post-Processing(후처리), Flow Visualization(유동 가시화), Flow Analysis(유동해석), OOP(Object-Oriented Programming; 객체지향프로그래밍), VTK(Visualization ToolKit), Class(클래스), GUI(Graphic User Interface; 그래픽 사용자 인터페이스)

Abstract

A post-processing program based on the OOP(Object-Oriented Programming) concept has been developed for flow visualization of the flow analysis code(PowerCFD) using unstructured cell-centered method. User-friendly GUI(Graphic User Interface) has been built on the base of MFC(Microsoft Foundation Class). The program is organized as modules by classes including those based on VTK(Visualization ToolKit)-library, and these classes are made to function through inheritance and cooperation which is an important and valuable OOP concept. The major functions of this post-processor program are introduced and demonstrated, which include mesh plot, contour plot, vector plot, surface plots, cut plot, clip plot, xy-plot and streamline plot as well as view manipulation (translation, rotation, scaling etc).

1. 서 론

복잡한 열/유체 유동현상을 수치해석방법으로 계산하고자 하는 경우, 계산영역의 형상에 대하여 격자를 구성하고 경계조건을 입력하는 전처리기(pre-processor), 유동현상을 지배하는 지배방정식을 적절한 수치해석방법을 사용하여 계산결과를 얻기 위한 솔버(solver), 솔버로부터 얻어진 결과를 이해 및 분석하기 용이하도록 컴퓨터그래픽으로 표현하여 주는 후처리기(post-processor)가 필요하게 된다. 이중 솔버를 위한 수치해석방법에 대한 연구는 국내 대학 및 연구소에서 활발하게 진행되어 비교적 많은 양의 연구결과가 축적되어 있으나, 전처리기와 후처리기에 대한 연구결과는 상대적으로 미흡하다.[1,2] 이와 같이 국내의 전처리 및 후처리 기술개발 수준이 외국에 비해 매우 낙후된 주요 원인은 CFD 해석 소프트웨어의 핵심엔진인 솔버에 대한 기술개발이 아직 완성되지 않았고, 또한 상업적으로 연계되지 못하였기 때문으로 사료된다. 따라서 후처리 분야는 최근까지 국내에서는 실험실 수준에서 간이방법으로 만들어 사용하던지, 주로 고가의 외국 상용 CFD 소프트웨어를 구입하여 사용하고 있다. 그러나 기존의 외국 상용 소프트웨어는 고가의 도입비용과 유지비용이 불가피하여 예산문제를 가중시키고 많은 설정과 기능들로 사용법을 익히

† 책임저자, 회원, 국민대학교 기계자동차공학부
E-mail : myong@kookmin.ac.kr
TEL : (02)910-5030 FAX : (02)910-4839
* 회원, 국민대학교 대학원 기계공학과

기도 어려운 문제점을 가지고 있다. 또한, 최근 해석결과를 더 유용하게 분석할 수 있는 각종 가시화(visualization) 기능에 대한 필요성이 커지면서, 후처리기에 대한 관심이 더욱 높아지고 있다.

최근 저자는 이러한 문제를 해결하고자 비정렬 셀 중심 방법(cell-centered method)에 기초한 3차원 유동해석솔버(PowerCFD)[5,6]를 개발하면서, 동시에 GUI(Graphic User Interface)환경하에서 OOP (Objected-Oriented Programming; 객체지향 프로그래밍) 개념에 기초하여 유동해석 결과를 가시화하고 교육용으로도 사용 가능한 후처리 프로그램(PowerCFD/PostP v1.0)을 일련의 연구[3,4]를 통해 개발하였다. 이 후처리 프로그램은 그래픽스 모델(graphics model)로 객체지향 개념으로 설계·구현된 VTK(Visualization Toolkit)[7~9]라는 3차원 그래픽 라이브러리를 사용함으로써 보다 강력하고 편리한 기능구현이 가능하고, 모듈(module)화를 이용한 단순한 구조의 프로그램으로 변경이나 확장에 유연한 후처리기이다. 또한, 이 후처리기는 유동 가시화에 있어서 입력된 데이터를 이미지로 표현하는 단순 그래픽 기능뿐만 아니라, 입력된 데이터를 필터 개념의 독립된 모듈로 구성함으로서 보다 자유로운 가시화 작업을 가능하게 하였다. 현재까지 개발된 주요 기능(특징)으로는 Vector Plot, Cut Plot, Clip Plot, Contour Plot, Streamline Plot, XY-Plot 및 Animation이 있다.

본 논문에서는 OOP 개념에 기초하여 개발된 후처리 프로그램(PowerCFD/ PostP v1.0)의 구조, 주요 기능 및 적용 예를 소개한다.

2. VTK 소개

일반적으로 3차원 그래픽을 컴퓨터로 구현하는 것은 용이한 작업이 아니다. 즉, 많은 사람들이 OpenGL 등을 공부하고 이를 이용해 3차원 그래픽을 구현하지만, OpenGL은 3차원 데이터를 모니터와 같은 하드웨어에 그려주는 기본적인 그래픽 라이브러리이기 때문에 거의 모든 것을 프로그래머가 작성해야 한다. 예를 들어 자동차의 충돌 시뮬레이션을 수행하고, 그 결과를 등고선 형태로 표현한다고 할 경우, 프로그래머는 등고선으로 출력하기 위해서 자동차를 구성하는 데이터에서 등고선으로 출력하는 알고리즘을 알아야 하고, 이것을 직접 프로그램으로 작성해야 한다. 또한 화면에 3차원 물체를 출력했는데 이를 확대하고 이동시키는 등의 연산을 하고 싶다면, 이것 역시 프로그래머가 직접 구현해야 한다. 즉, OpenGL을 직접 사용하는 것은 Windows 프로그래밍을 MFC를 사용하지 않고 SDK를 이용하여 프로그램을 작성하는 것과 비슷하다.

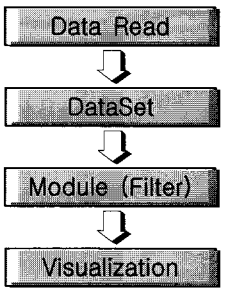

Fig. 1 Flow chart of present post-processing program(PowerCFD/PostP v1.0)

본 연구에서 사용한 VTK[7~9]는 변수와 함수를 결합한 클래스(class)와 클래스간의 파생과 상속(inheritance) 등의 개념을 제공하므로, 효율적이고 강력한 후처리 프로그램 개발을 가능하게 하는 객체지향 개념으로 설계·구현된 3차원 그래픽 라이브러리이다. 또한 복잡한 그래픽스 기능들에 관한 대부분의 알고리즘은 이미 구현되어 모듈로 만들어져 있기 때문에, 유연한 확장성과 함께 쉽게 사용할 수 있는 장점이 있다. 즉, 컴퓨터 그래픽과 가시화에 대한 기본 개념을 알고 있으면, 누구나 비교적 용이하게 후처리 프로그래밍이 가능하다.

3. 프로그램의 구조 설계

3.1 개발 목표

본 프로그램의 개발 목표는 다음과 같다.

첫째, 특화된 가시화 프로그램 개발로, 프로그램 사용방법을 익히는데 들어가는 시간을 최소한으로 줄여주기 위해 외국 상용 CFD 소프트웨어들의 가시화 기능들 중에서 많이 쓰이면서도 중요한 기능을 구현하여 데이터 가시화 및 분석용으로 활용하는 것이다.

둘째, 사용자에게 친숙한 환경을 제공해줄 수 있는 편리한 GUI 환경을 구축하는 것으로, 이것은 사용자들을 확보하고 프로그램을 성장시킬 수

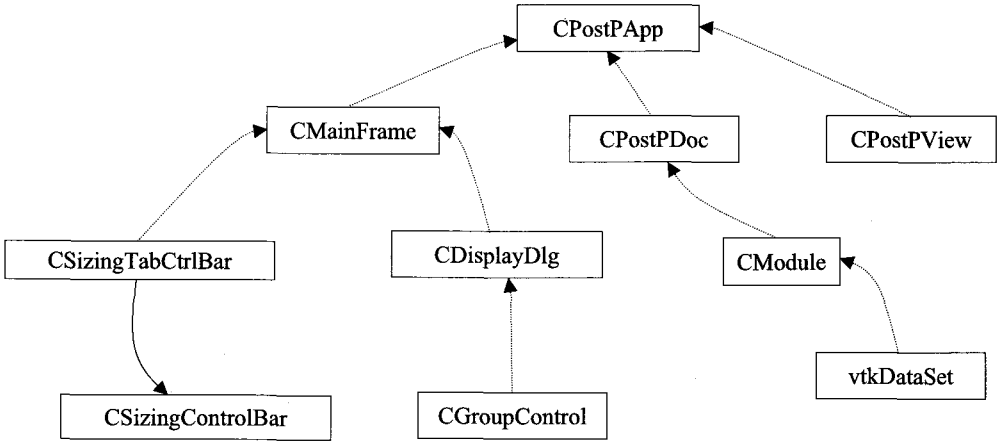

Fig. 2 Class inheritance and collaboration diagram

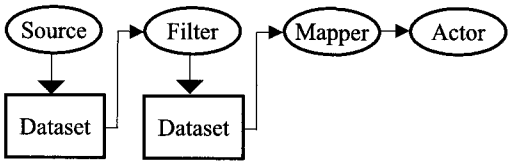

Fig. 3 Imaging model pipeline

있는 가장 기본적이면서 중요한 사항이다.

셋째, 확장성 및 관리의 용이성을 위해 OOP 개념에 기초하여 프로그램 구조를 설계하는 것이다.

3.2 프로그램의 순서도

본 연구를 통해 개발된 가시화 프로그램이 실행되는 기본절차를 간단하게 표현하면 Fig. 1과 같다. 즉, 데이터를 읽은 후 메모리를 동적으로 생성하여 데이터를 저장할 수 있는 클래스 객체(object)에 데이터를 저장하고, 저장된 데이터는 필터(filter)라는 독립된 모듈에 의해 가공되어 화면에 출력되는 시스템이다.

3.3 클래스

Fig. 2는 본 연구를 통해 개발된 프로그램의 클래스 상속(inheritance) 및 협력(collaboration)관계를 도표를 나타낸 것으로 실선은 상속관계를, 점선은 협력관계를 각각 나타내고 있다. 그림에서 각각의 박스들은 후처리 프로그램의 주된 클래스를 의미하며, 그래픽스 모델로 사용되는 VTK 클래스는 너무 많은 수가 사용되므로 vtkDataSet을 제외하고는 그림에 나타내지 않았다.

3.4 그래픽스 모델

본 프로그램에서 사용하는 그래픽스 모델은 VTK에서 제공하는 클래스를 주로 이용하고 있다. 이것은 OpenGL의 렌더링 라이브러리 개념보다 높은 레벨로, 이해하고 사용하기 쉬운 장점이 있기 때문이다. 참고로 본 후처리 프로그램을 개발하는데 있어 적극적으로 응용한 기본 객체(basic object)는 그래픽스 모델에서 윈도우(window)와 같은 GUI 환경에 기반을 둔 아래의 8가지 기본 객체이다.

1. Render Window : 장치에 보여 지는 윈도우를 관리한다. 물체를 표현하기 위해 하나의 윈도우에 여러 개의 랜더러(renderer)가 적용되기도 한다.
2. Renderer : 렌더링되는 Light, 카메라(camera), Actor를 표현한다.
3. Light : Actor가 보이도록 빛을 발한다.
4. 카메라 : 시점, 초점, 카메라의 특징 등을 계산한다.
5. Actor : Renderer에 의해 구현되는 모든 것.
6. Property : 색상, 빛, 질감, 스타일처럼 렌더링 시 Actor로부터 표현되는 속성.
7. Mapper : Lookup Table을 통해 매핑(mapping)된 물체를 표현한다. 하나 이상의 물체가 같은 Mapper를 사용할 수 있다.

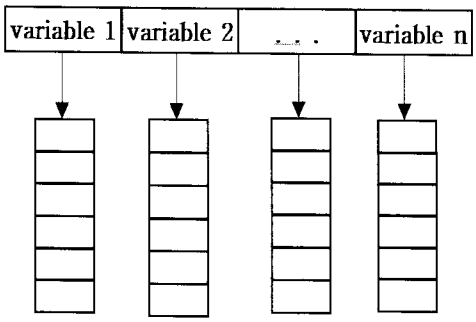

Fig. 4 Structure of dataset

8. Transform : Actors, 카메라, Lights의 위치와 방향 값을 가진다.

3.5 Imaging Model

Fig. 3에는 본 후처리 프로그램에서 채택한 imaging model pipeline을 나타냈다. 그림에서 보는 바와 같이 본 연구에서는 해석 솔버로부터 얻어진 데이터를 구조화된 Dataset으로 생성시킨 후, 이것을 목적에 맞게 가공시키는 특수한 기능의 필터를 거쳐 하나의 기능을 수행시키기 위한 새로운 출력 Dataset 모듈을 생성시키는 방법을 사용하였다. 즉, Dataset을 대상으로 하는 처리과정은 항상 입력(input)과 출력(output)의 독립적인 Dataset 객체를 가지며, 필터의 결과물인 출력 Dataset은 Mapper를 거쳐 Actor로 가시화되도록 하였다. 이것은 본 연구에서 개발된 프로그램의 특징 중의 하나로 입력된 데이터를 필터 개념의 독립된 모듈로 구성함으로서 보다 자유로운 가시화 작업을 가능하게 하기 위해서이다.

4. 프로그램의 요소 설계

4.1 Dataset

Dataset은 하나의 격자 데이터와 계산결과인 다수의 종속변수들을 동적으로 메모리를 할당해 저장한다. 즉 Dataset은 파일형태로 입력된 데이터를 모두 저장한다. 저장된 종속변수는 서로 구분지울 수 있는 변수명과 함께 저장되어 있으며, 특정 변수를 사용할 때는 SetActiveScalars(변수명) 또는 SetActiveVectors(변수명)로 특정 변수를 활성화하여 사용한다. 참고로 Fig. 4에 데이터가 저장되는 구조를 나타내었다.

4.2 필터(filter)

CModule 클래스인 필터는 본 프로그램의 특징 중의 하나로 vtkDataSet과 협력관계에 있어 저장된 Dataset을 목적에 맞게 가공하여 새로운 출력 Dataset을 생성한다. 구체적으로 Contour, Cutting, CellDataToPointData, Clip, Streamline, XY-Plot, Glyph 등이 있다. 목적에 맞게 생성된 CModule 클래스의 객체는 한가지의 기능을 수행하며, 멤버변수로 Dataset을 가지고 있어 객체들 간에 독립성을 유지한다. 즉 처음 생성된 CModule 클래스의 객체는 완전한 Dataset을 가지며, 이후 필터를 거쳐서 생성된 객체들은 필터링(filtering)의 결과물인 독립적인 Dataset을 가지므로 객체들 간에 독립성을 유지할 수 있다. 또한 필터를 한번 통과한 Dataset을 입력데이터로 해서 CModule 클래스의 객체를 생성할 수도 있다. 즉 필터들 간의 중첩을 허용하고 있어 계산결과물인 데이터의 필요한 부분을 보다 자유롭게 가시화할 수 있다.

Contour

표면 등고선(surface contour)을 구현하기 위해 사용된 알고리즘은 등고선을 그리기 위한 방법으로 많이 사용되고 있는 Marching Square 알고리즘이다.[7] 이 알고리즘의 원리는 사변형 형태의 모든 셀에 대하여 각 셀의 변에서 동일한 값을 꼭짓점으로부터 보간하여 찾아내고 선으로 이어 주는 것이다. 하나의 셀에서 Contour Line을 만들 수 있는 경우는 16가지이지만 대칭성을 고려하면 네 가지로 줄일 수 있다.[7]

Iso-Surface를 구현하기 위한 알고리즘은 널리 사용되고 있는 Marching Cubes 방법이다.[7] 이 방법은 삼각형 조각을 이어 붙여 면을 만드는 방법으로써 육면체의 꼭짓점의 선택여부에 따라 256가지의 경우의 수가 생길 수 있지만, Contour Line에서와 비슷하게 대칭성을 고려하면 14가지 경우로 축약시킬 수 있다.[7]

유선(streamline)

유선을 구현하기 위한 방법은 먼저 데이터의 물리적인 공간상에서 시작점을 임의로 정하고 작은 시간간격 후의 위치를 계산하여 시작점으로부터 매 시간간격 후의 점들을 연결하는 것이다.

Cutting

Dataset을 임의의 평면으로 자르고 각 데이터 값을 보간 하여 평면상의 값을 표현하는 것을 Cutting 또는 Data Cutting이라 한다. Cutting된

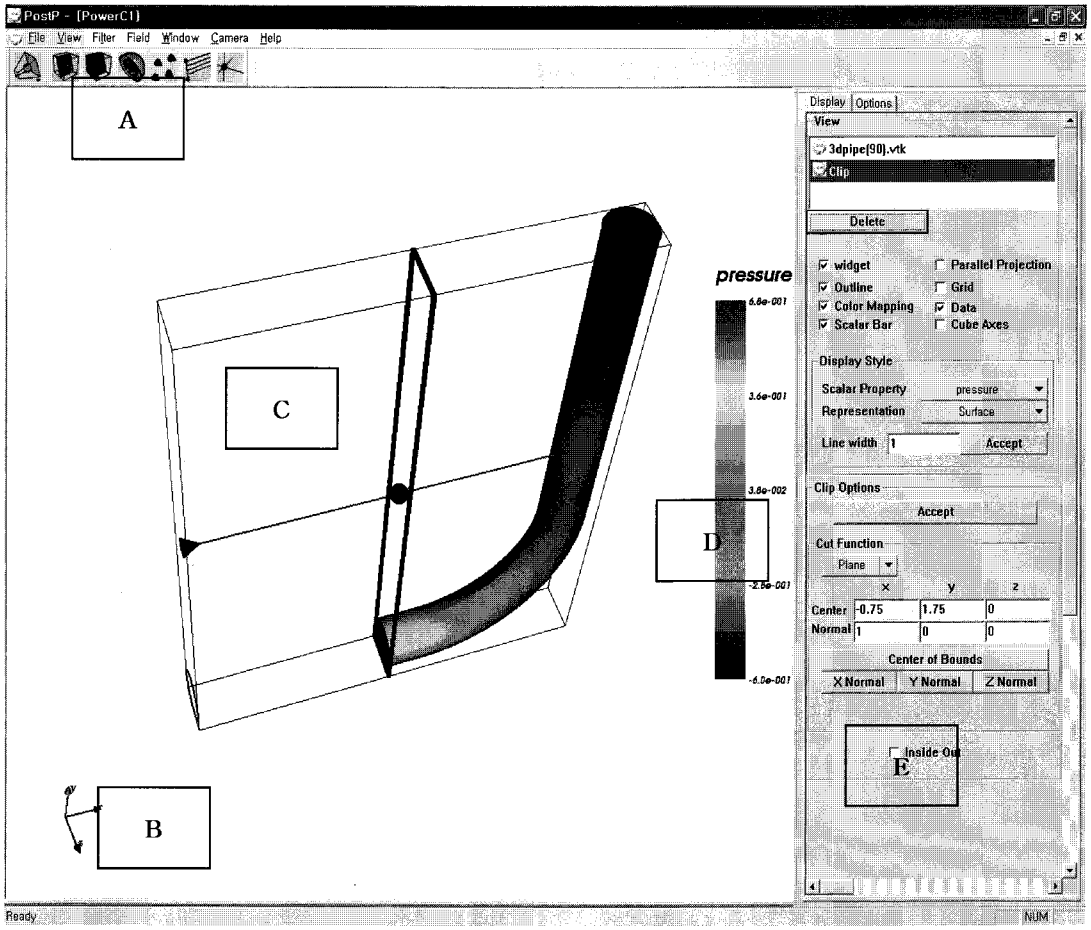

Fig. 5 Main window of GUI

평면상의 값은 Color Mapping되어 화면상에 표현된다.

Glyph

벡터(vector) 데이터를 방향성과 크기를 가진 Arrow-Source를 사용해 표현한다. Arrow-Source는 각 점으로부터 시작되며 3가지 성분의 데이터로 방향과 크기가 결정된다. 또 Arrow-Source의 크기별로 색깔을 구분하여 컬러 바(color bar)와 매칭(matching)시키도록 하였다.

4.3 속성 변환 버튼 및 컨트롤

Fig. 2에 나타낸 CDisplayDlg 클래스는 CModule 클래스와 협력관계에 있는 클래스로 CModule 클래스의 속성 즉 필터의 종류에 맞게 버튼 및 여러 가지 컨트롤 등을 바꿔주는 기능을 한다. 참고로 본 프로그램의 또 다른 특징 중에 하나는 일반 상용 CFD 코드의 후처리 장치와 달리 각각의 기능을 구현하기 위한 메뉴가 별도의 공간을 필요로 하지 않는다는 것으로, 이 기능을 수행하는 것이 CDisplayDlg 클래스이다. 예를 들어 Clip 필터를 수행시켜 Clip 모듈을 생성해내면 Main View 오른쪽에 위치한 창에는 Clipping 기능과 관련된 여러 가지 컨트롤이 나타나지만, Cut 모듈을 생성해내면 이번에는 Cutting 기능과 관련된 컨트롤이 나타나 여러 가지 가시화 작업을 용이하게 수행할 수 있다. 이는 대부분의 상용 CFD 코드에서 기능마다 각각의 대화상자(dialog)가 생성되어 Graphic Viewport를 가려서 대화상자를 매번 이동해야하는 번거로움을 덜어주는 매우 편리한 기능이다.

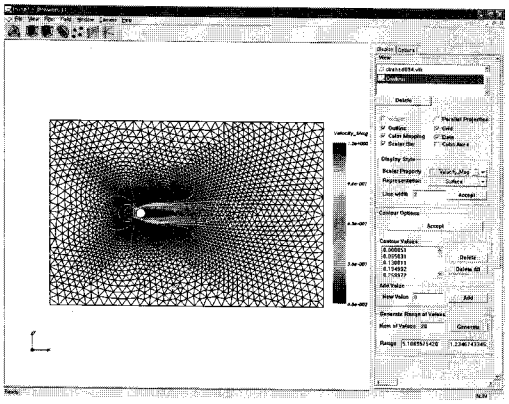

Fig. 6 Examples of mesh and contour plot

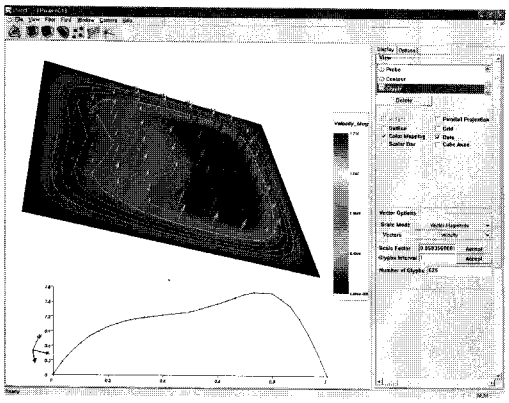

Fig. 7 Examples of contour and vector and XY-plot

5. 프로그램의 구성 및 기능 적용 예

5.1 GUI 구성

본 연구를 통하여 개발된 가시화 프로그램의 화면구성은 Fig. 5와 같이 이루어져 있으며 각 영역의 주된 기능은 다음과 같다.

A) ToolBar: ToolBar는 자주 사용하는 기능들을 간단한 그림을 사용해 표시한 버튼형식으로 구성되어있다. 사용을 위한 별도의 교육이 없더라도 ToolBar에 그려진 기능을 나타낸 그림과 ToolTip을 보면 누구나 손쉽게 기능을 구현할 수 있다.

B) Axes: 격자의 좌표축을 나타내며 마우스로 끌기를 하면 위치를 이동할 수 있다.

C) Graphic Viewport: 3차원 그래픽 처리 영역.

D) Color Bar: Graphic Viewport에 나타난 데이터가 Color Mapping되어 있을 경우, 각 색깔들의 정도를 숫자로 나타낸다. Check Button으로 On/Off도 가능하다.

E) Property Sheet: 각 필터들의 세부 설정을 조정하는 컨트롤들을 생성한다.

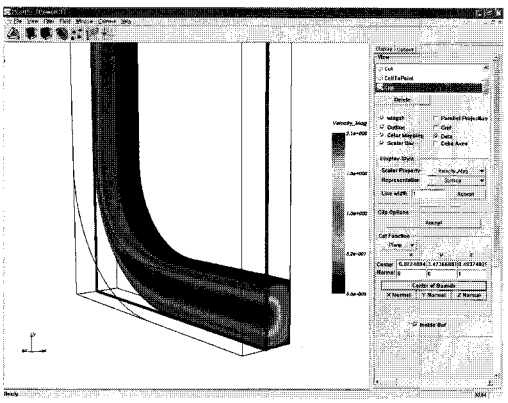

Fig. 8 Example of clip plot

5.2 Mesh Plot

격자의 형태를 관찰하기 위해 모든 격자점을 연결하여 가시화하는 기능으로, 기본적으로 비정렬격자계를 대상으로 하였기 때문에 정렬 및 비정렬 격자계 모두 가시화하는 것이 가능하다. 격자의 외곽선(outline)만을 그려주는 기능도 선택가능하며, on/off 설정은 Check 버튼 형태로 되어 있어 손쉽게 구현 할 수 있도록 하였다. Fig. 6은 2차원 데이터에서 메쉬(mesh)와 등고선(contour)를 동시에 표현한 것이다.

5.3 Contour Plot

Contour Plot 기능은 3차원 데이터에 대해서는 Iso-surface를 구현하고, 2차원 데이터에 대해서는 Contour-line을 구현한다. 본 프로그램은 등고선 값의 최대값을 적색으로 최소값을 청색으로 표현하고 있다. 또한 사용자의 편의를 위해 등고선 개수를 지정해서 자동으로 생성하거나 하나씩 수동으로 생성할 수 있으며, 메뉴창의 Contour Values List에서 등고선 값을 추가하거나 삭제할 수도 있도록 하였다. 이와 함께 현재 Contour Plot으로 표현된 변수를 변경할 경우 자동으로 변경된 변수의 최대값과 최소값을 등고선 개수만큼 등간격으로 나누어 화면에 출력하도록 만들어져 있기 때문에 사용자는 자유롭게

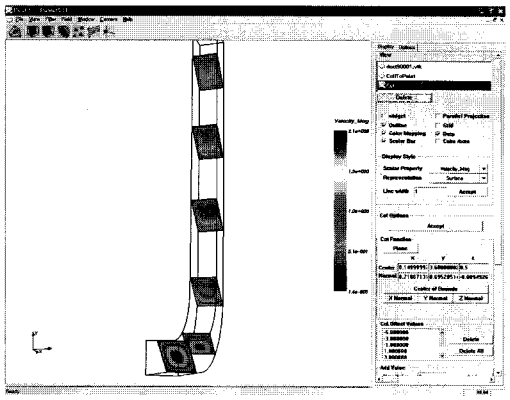

Fig. 9 Example of cut plot

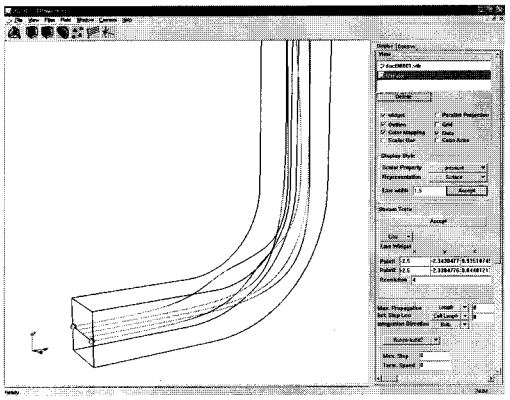

Fig. 10 Example of streamline plot

변수를 변경하여 볼 수 있다.

5.4 Vector Plot

Vector Plot은 Fig. 7과 같이 다른 모듈과 중첩하여 표현될 수 있으며, Contour Plot의 경우와 같이 변수를 자유롭게 변경할 수 있다. 벡터의 크기가 너무 작거나 클 경우 사용자가 임의로 조정할 수 있다. 또 격자점의 수가 너무 많아 컴퓨터의 사양에 따라 Arrow-Source를 모든 격자점에 생성하지 못하던지, 그림상으로 구별할 수 없을 정도로 조밀한 경우 표현 가능한 범위 내에서 성기게 생성할 수도 있도록 하였다.

5.5 Clip Plot

Clip Plot은 Fig. 8에 나타낸 것과 같이 임의의 평면으로 절단된 나머지 데이터만을 화면에 나타내는 기능이다. Inside Out 기능이 있어 반대편의 데이터만을 볼 수도 있다. 임의의 평면은 Plane Widget이라고 하는 도구에 의해서 자유롭게 설정할 수 있게 하였다. 또한 마우스로 화살표(arrow0의 머리나 꼬리를 잡아 돌리면 평면도 따라서 회전하고, 평면을 이루고 있는 외각의 굵은 튜브를 마우스로 잡아끌면 평행이동을 할 수도 있게 만들어져 있다.

5.6 Cut Plot

Cut Plot은 Fig. 9와 같이 3차원으로 표현된 데이터를 마우스를 사용해 임으로 설정 가능한 평면을 사용하여 자르고 단면을 볼 수 있는 기능이다. 사용자 편의를 위해 본 연구에서는 몇 개의 Offset 값을 입력함으로써 동시에 여러 개의 Cutting 단면을 생성할 수도 있도록 하였다.

5.7 XY-Plot

Fig 7은 본 프로그램의 특징 중의 하나인 필터의 중첩효과를 나타낸 것으로, Cutting기능을 사용하여 생성된 데이터에 Contour 기능을 사용하여 등고선으로 표현하고, 다시 XY-Plot 기능을 사용하여 특정 Line상의 값을 그래프로 나타낸 것이다. 데이터를 가로지르는 Line은 Line Widget이라고 하는데 양끝에 마우스로 끌기가 가능한 구가 달려있어 손쉽게 Line의 위치를 변경할 수 있다.

5.8 Streamline Plot

Fig. 10은 Streamline plot을 구현한 예이다. Line Widget상의 점들이 Streamline의 시작점으로 사용된다. 시작점의 개수는 메뉴판에서 조정할 수 있으며 시작점은 Line Widget을 사용하였기 때문에 마우스로 손쉽게 변경할 수 있다.

5.9 Animation

비정상 상태의 문제를 해석할 경우 각 시간대에 따라 파일이 생성되는데 이 파일들을 모두 읽어 Dataset을 생성함으로써 시간에 따른 종속변수(vorticity, temperature, pressure, velocity 등)의 변화를 연속된 그림으로 살펴볼 수 있다.

6. 결 론

본 논문에서는 GUI 환경하에서 OOP(객체지향 프로그래밍) 개념에 기초하여 개발된 3차원 유동

가시화를 위한 후처리 프로그램(PowerCFD/ PostP v1.0)의 구조, 주요 기능 및 적용 예를 소개하였다. 이 후처리 프로그램은 그래픽스 모델로 객체지향기법으로 설계·구현된 VTK라는 3차원 그래픽 라이브러리를 사용함으로써 보다 강력하고 편리한 기능구현이 가능하고, 모듈화를 이용한 단순한 구조의 프로그램으로 변경이나 확장에 유연한 후처리기이다. 또한, 이 후처리기는 유동 가시화에 있어서 입력된 데이터를 이미지로 표현하는 단순 그래픽 기능뿐만 아니라, 입력된 데이터를 필터 개념의 독립된 모듈로 구성함으로서 사용자로 하여금 보다 자유로운 가시화 작업을 가능하게 하였다.

향후, 이 후처리 프로그램이 후처리기에 대한 국내 연구자들의 관심을 고취시켜 국내에서 경쟁력 있는 후처리기가 개발되고, 또한 국내 대학에서 CFD 교육을 위한 후처리기로 개발될 수 있도록, 다양한 실험적 기법들을 채용하고 테스트할 예정이다.

후 기

본 연구는 서울시 산학연 협력사업(2005년도 신기술연구개발 지원 사업)의 연구비를 지원받아 수행된 연구이다.

참고문헌

(1) Sah, J. Y. and Huh, J. S., 2003, "Code Development for Two-Dimensional Flow Visualization," *KSCFE, J. Computational Fluids Engineering*, Vol. 8, No. 1, pp. 30~37.

(2) Na, J. S., Kim, K. Y. and Kim, B. S., 2004, "Study on a Post-Processing Program for Flow Analysis based on the Object-Oriented Programming Concept," *KSCFE, J. Computational Fluids Engineering*, Vol. 9, No. 2, pp. 1~10.

(3) Myong, H. K. and Ahn, J. K., 2006, "Development of a Post-Processing Program for Analysis based on the Objected-Oriented Programming," *Proc. of the KSME 2006 Spring Annual Meeting*, June 7-9, Jeju, Korea, pp. 777~782.

(4) Myong, H. K. and Ahn, J. K., 2006, "Development of a Post-Processing Program for Flow Analysis," *Proc. of the 4th National Congress on Fluid Engineering*, Aug. 23-25, Kyungju, Korea, pp.901~904.

(5) Myong, H. K. and Kim, J., 2005, "Development of 3D Flow Analysis Code using Unstructured Grid System(1st Report, Numerical Method)," *Trans. of the KSME(B)*, Vol. 29, No. 9, pp. 1049~1056.

(6) Myong, H. K., Kim, J. and Kim, J. E., 2005, "Development of 3D Flow Analysis Code using Unstructured Grid System(2nd Report, Code's Performance Evaluation)," *Trans. of the KSME(B)*, Vol. 29, No. 9, pp. 1057~1064.

(7) http://kitware.com

(8) The VTK User's Guide, ver. 4.2, 2003, Kitware, Inc.

(9) The Visualization Toolkit, An Object-Oriented Approach to 3D Graphics, 3rd ed., 2002, Kitware, Inc.