ISBN 978-89-6211-401-0 98560

GIVI: GLOVE 시스템을 위한 통합 VR 인터페이스의 설계와 구현

허 영 주 (popea@kisti.re.kr)

한 국 과 학 기 술 정 보 연 구 원
Korea Institute of Science & Technology Information

목차

그림 차례

1. 서론

컴퓨터에서 수행한 시뮬레이션의 결과는 일반적으로 수치 데이터로 나타나게 되며, 이런 수치 데이터는 가시화 과정을 거쳐서 사람이 직관적으로 쉽게 이해하고 분석할 수 있는 형태를 갖추게 된다.

최근에는 고성능 컴퓨터(HPC: High Performance Computer)의 발달로 인해 데이터의 용량과 복잡도가 증가하는 추세에 있으며, 이런 복잡한 데이터를 해석하는 데는 그에 상응하는 수준의 가시화 기술과 고해상도의 디스플레이 장치가 필요하다. 그러나 이런 기기들을 이용해서 복잡한 시뮬레이션과 가시화 과정을 제어하기 위해서는 해당 분야에 대한 전문적인 지식 뿐만 아니라 고성능 컴퓨터 및 고해상도 디스플레이 장치에 대한 전반적인 이해와 지식이 요구된다.

이런 기기에 대한 기반지식이 필요한 가장 큰 이유는 고해상도 디스플레이 장치에서 이용할 수 있는, 사용자에게 친화적인 사용자 인터페이스가 없다는 데 있다. 따라서 이런 대형 가시화 시스템을 제어할 수 있는 사용자 인터페이스의 개발이 시급한 형편이다. 게다가 이런 대형 기기에서는 평면 모니터에서 수행하던 단순한 인터페이스보다는 이보다 한단계 더 발전된 형태의 인터페이스가 요구된다. 이런 사용자 요구에 대한 해결 방법으로는 현재 VR 기술이 적용되고 있으며, 이런 VR 기술은 대형 디스플레이 시스템에 대한 인터페이스를 제공할 수 있는, 유일한 해결책이기도 하다.

본 문서에서는 시뮬레이션 데이터, 특히 로터 동역학 분야의 시뮬레이션 데이터를 가상현실 환경에서 가시화하고 제어하는 통합 가시화 인터페이스를 설명할 것이다. 이 인터페이스는 고해상도 디스플레이 장치를 이용한 가상현실 환경에서 데이터를 실시간으로 상호작용을 통해 분석하는 데 필요한 기반 환경을 제공할 것이다.

본 문서에서는 우선 로터 동역학 분야의 데이터를 가상현실 환경에서 제어하는 GLOVE 프레임워크를 설명하고 가상현실 환경에서 사용하는 인터페이스에 대해 설명한 뒤, GLOVE의 인터페이스에 대해 설명하기로 한다.

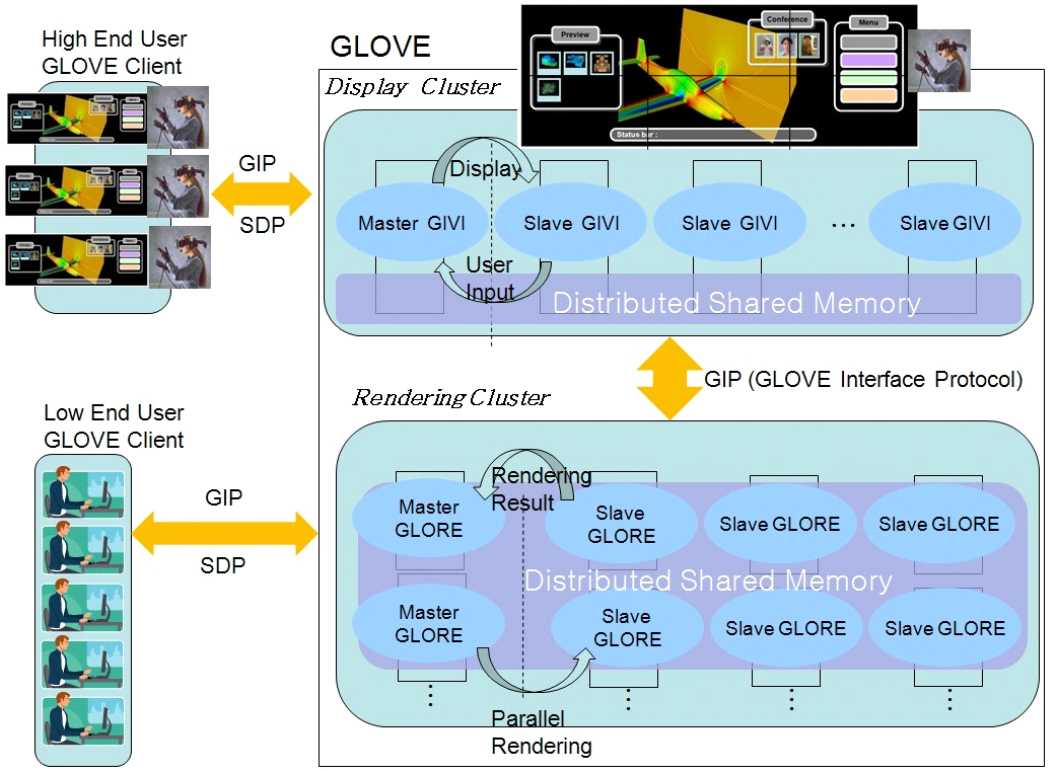
2. GLOVE (GLObal Virtual Environment for collaborative research)

GLOVE는 고성능 컴퓨팅 환경에서 대용량 시뮬레이션 데이터를 효과적으로 분석하고 가시화할 수 있게 해주는 통합 가시화 프레임워크로, 현재는 로터 동역학 시뮬레이션 데이터 분석을 타겟으로 개발되고 있다.

GLOVE의 기본 구조는 [그림 2-1]과 같다.

GLOVE는 통합 사용자 인터페이스인 GIVI(GLOVE Integrated Visualization Interface)와 데이터 가공 및 렌더링을 담당하는 GLORE(GLOVE Rendering Engine), 이 2부분으로 나뉘어져 있다.

GIVI는 가상현실 입/출력 장치를 총괄하며, 사용자로부터 입력을 받아서 GLORE에 전달하고 GLORE의 실행 결과를 전달받아 화면에 출력하는 역할을 수행한다. 반면, GLORE는 대용량 데이터의 가공 및 렌더링을 위한 GIVI의 서브시스템이다.



[그림 2-1] GLOVE 구조

통상적으로 GLOVE는 GLORE와 GIVI가 서로 독립적으로 운영되는 클러스터 환경에서 실행되는데, 이런 구조는 GLOVE의 인터페이스(GIVI) 부분과 핵심 렌더링 기능(GLORE)을 물리적으로 분리할 수 있기 때문에 충분한 그래픽 처리능력을 갖추지 못한 시스템을 보유하고 있는 사용자의 경우에 매우 유용한 구조라고 볼 수 있다. 즉, 사용자가 보유하고 있는 시스템에서는 GIVI만 실행하고 원격지의 고성능 컴퓨터에서 실행되는 GLORE를 이용해서 대용량 데이터를 처리할 수 있는 것이다. 이렇게 분산된 시스템 구조에서는 데이터의 공유를 위한 효율적인 전송이 반드시 필요하다. GLOVE는 GIVI와 GLORE 사이의 효율적인 데이터 전송을 위해 초고속 네트워크 기반의 분산 캐시 메커니즘을 이용한다. 이 메커니즘은 디스크 입출력 횟수를 줄이고, 서로 독립적으로 운영되는 클러스터 사이의 병렬 데이터 전송을 가능케 한다.

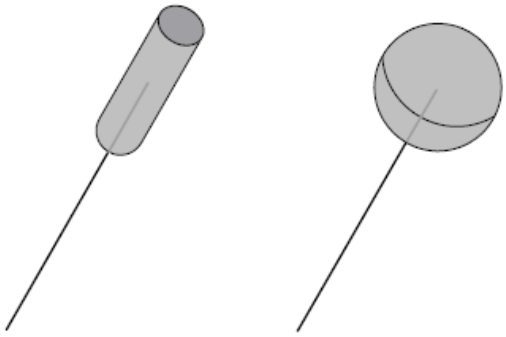
즉, GLORE와 GIVI로 구성된 GLOVE는 범용적인 가시화를 위한 프레임워크 위에 각 애플리케이션의 특성을 반영하는 인터페이스를 지원하는 것으로, 현재는 로터 동역학 시뮬레이션 분야를 타겟 분야로 개발이 진행중이다.

3. 가상현실 환경에서 사용되는 3차원 인터페이스

현재 가상현실 환경에서 사용되고 있는 3차원 인터페이스는 매우 다양하다. 3장에서는 이런 다양한 인터페이스 및 위젯에 대해 설명하기로 한다.

1) 3D 커서 리젼(3D Cursor Region)

3D 커서 리젼([그림 3-1])은 3차원 공간에서 선택을 단순화하기 위한 공간으로, 리젼 안에 들어가는 오브젝트는 3D 커서 리젼에 의해 선택된 오브젝트다. 3D 커서 리젼은 구 형태와 원통 형태의 2가지가 있으며, 사용자의 손으로부터의 거리와 리젼 크기를 파라미터로 지정할 수 있다.



[그림 3-1] 3D 커서 리젼

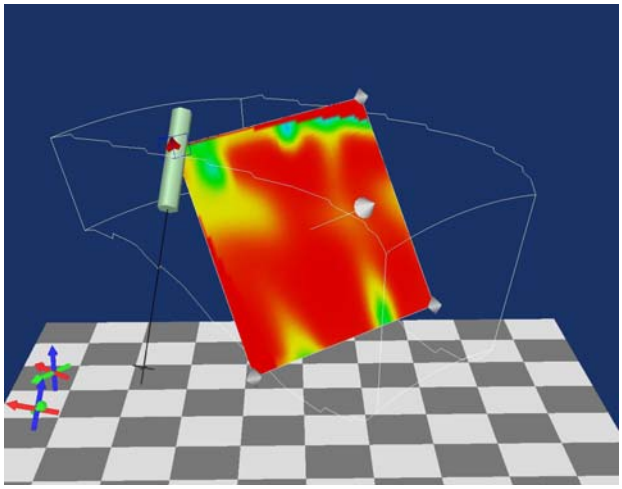
2) 3차원 위젯

데이터 가시화와 관련된 모든 3차원 인터랙션은 3차원 위젯을 조작해서 이뤄지는 것이 사용자 친화성이 높다.

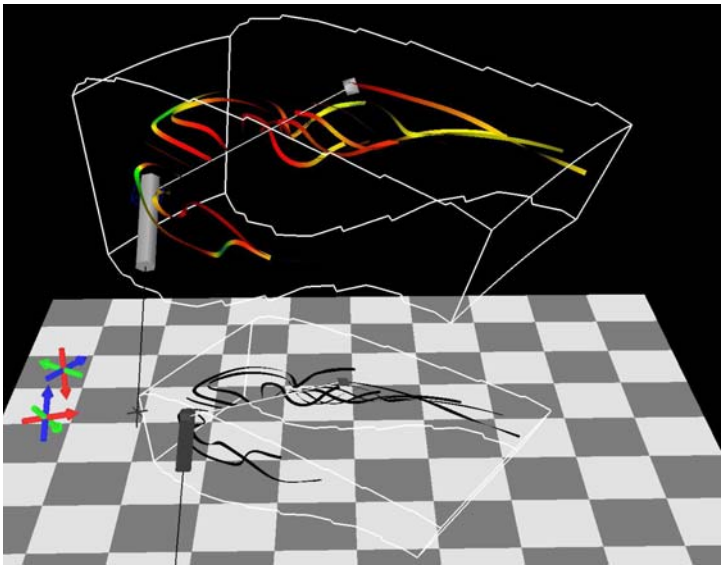
3차원 위젯은 [그림 3-2]와 같이 사용자가 핸들을 이용해서 직접 조작함으로써 가시화 데이터를 제어하는 형태로 이뤄지며, plane widget, point widget 등, 다양한 형태의 위젯을 이용함으로써 다양한 형태의 동작을 구현하는 것이 가능하다.

3차원 위젯을 구현하기 위해서는 일단 위젯의 기하 정보에 대한 3차원적인 조작이 필수적이며, 트랙커로부터 입력받은 공간 데이터를 통해 위젯의 기하정보를 조작하는 기능이 필요하다. 그런 다음에는 위젯과 관련된 액션을 구현함으로써 직접적으로 데이터를 제어하는 기능을 제공하면 된다.

[그림 3-3]의 3차원 위젯의 예를 설명해 보겠다. [그림 3-3]의 위



[그림 3-2] 3D 위젯

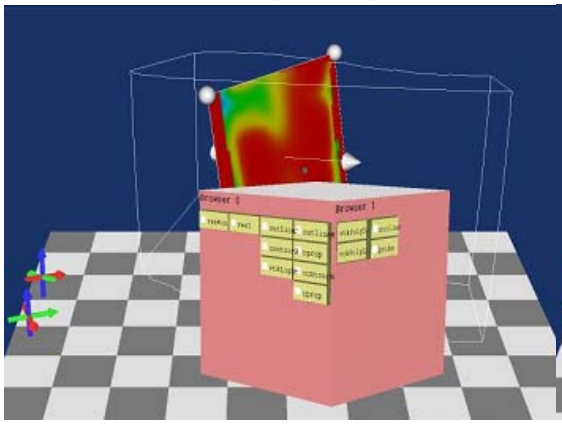


[그림 3-3] VR-VTK의 스테이지와 그림자

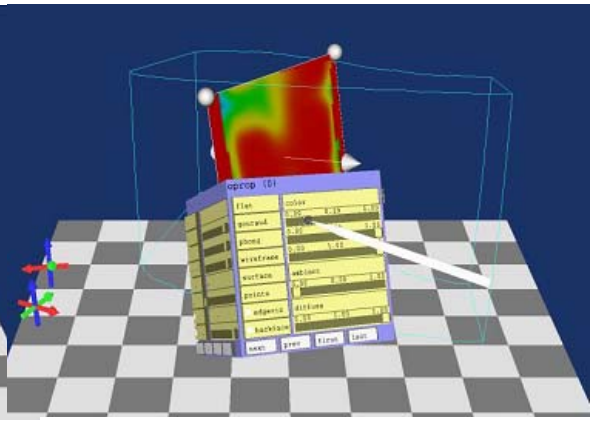
젯은 슬라이스 평면을 제어하는 평면 형태의 위젯이다. 이 위젯은 크기를 조절하는데 모서리에 붙어있는 핸들을 사용하고, 화살표 모양의 핸들을 사용해서 평면의 각도를 조절한다. 또, 평면을 잡고 조작하면 평면 자체를 움직일 수도 있다.

이외에도 World in hand, Motion parallax, 스테이지, 그림자 등의 다양한 기능을 제공함으로써 사용자의 데이터에 대한 인지도를 높일 수 있다.

3차원 인터페이스의 기능 중에서 빼놓을 수 없는 부분은 특히 사용자 인터페이스와 관련된 기능이다. 사용자 인터페이스는 가시화 메소드를 사용 가능한 상태로 만들거나 파라미터 설정, 오브젝트를 위한 속성 설정, 컬러 테이블을 수정하는 등, 애플리케이션을 직접 제어하는 역할을 담당한다.



[그림 3-4] 파이프라인 인터페이스



[그림 3-5] 오브젝트 인터페이스

일반적으로 사용자 인터페이스 툴킷의 경우에는 버튼, 메뉴, 슬라이더와 같은 2차원 위젯을 포함한다. 또, 이런 2차원 위젯들을 구성하는 방식에 따라 다양한 사용자 인터페이스를 구성할 수도 있다. [그림 3-4]와 [그림 3-5]는 이런 사용자 인터페이스를 보여주는 사례다.

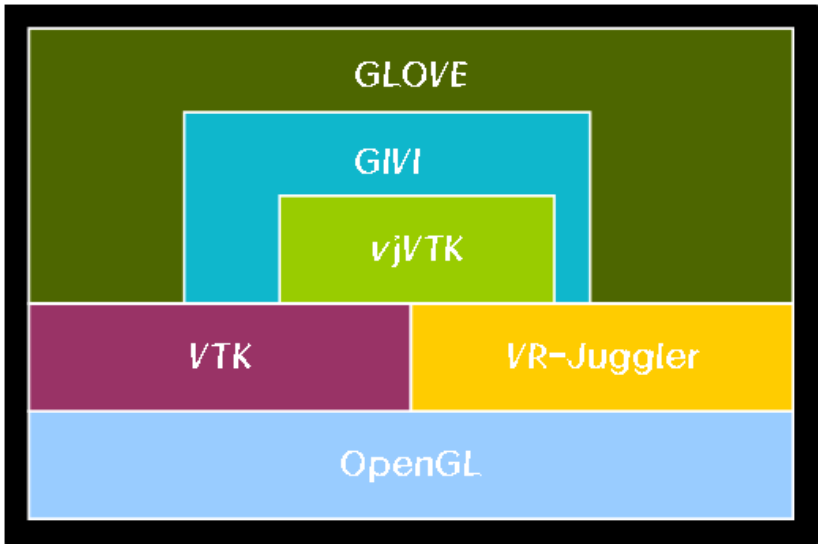
[그림 3-4]와 [그림 3-5]의 인터페이스는 큐브에 2차원 인터페이스를 부착함으로써 하나 이상의 면에 연결된 그래프의 형태로 인터페이스를 구성한 것이다. 이렇게 함으로써 사용자가 필요로 하는 기능을 제공하면서 각 오브젝트에 대한 속성 및 메소드를 제어할 수 있다.

4. GIVI: GLOVE Integrated Visualization Interface

GIVI는 GLOVE에 대한 통합 사용자 인터페이스다. 4장에서는 GIVI의 구조 및 기능, 그리고 각 클래스에 대해 알아보기로 한다.

가. 구조 및 기능

GIVI는 VTK와 VRJuggler를 사용한다. 시뮬레이션의 결과 데이터를 가시화하는 방법은 매우 다양하며, 때로는 복잡한 가시화 알고리즘을 필요로 한다. GIVI에서는 VTK(Visualization ToolKit)가 바로 이런 역할을 수행한다. VTK는 3차원 컴퓨터 그래픽스, 이미지 처리 및 가시화에 주로 사용되는 공개 소프트웨어 라이브러리로 스칼라, 벡터, 텐서, 텍스쳐 및 볼륨 데이터를 표현할 수 있는 자료구조뿐만 아니라 implicit modeling, polygon reduction, mesh smoothing, 컷팅(cutting), 컨투어링(contouring)과 같은 고급 모델링 기법도 제공한다. 또, 실제적으로도 CFD, 의료, 화학, 구조역학 등 다양한 분야에서 널리 사용되고 있으므로 과학 데이터 가시화에 있어 매우 유용한 툴임은 분명하다. 하지만 VTK는 고해상도 디스플레이 장치 출력이나 가상현실 환경에 관련된 기능은 전혀 지원하지 않는다. 그래서 GIVI는 가상현실 환경을 제공하는 프레임워크인 VRJuggler를 이용해서 VTK의 렌더링 결과를 가상현실 환경에서 디스플레이할 수 있게 했다. VRJuggler는 개발상 자유도가 매우 높은 미들웨어로, 매우 다양한 형태의 기기를 지원한다. GIVI는 이런 VRJuggler와 VTK를 연동함으로써 키보드와 마우스뿐만 아니라 IS-900, CAVE, 타일형 디스플레이장치



[그림 4-1] GIVI 프레임워크

같이 특수한 형태의 가상현실 입출력 장치를 지원하게 함으로써 다양한 형태의 기기를 지원할 수 있게 했다. 이에 따라 사용자는 PC부터 CAVE나 대형 타일형 디스플레이 장치에 이르기까지, 다양한 가시화 시스템에서 동일한 형태로 실행하는 것이 가능하다.

VRJuggler와 VTK의 연동에는 vjVTK를 사용했다. vjVTK는 주로 렌더링과 관련된 부분에 관계하는 미들웨어로, VTK의 렌더러 관련 기능을 가상현실 환경으로 출력할 수 있게 해준다. GLOVE의 GIVI 인터페이스 프레임워크는 [그림 4-1]과 같다.

GIVI는 VRJuggler 애플리케이션의 동작 매커니즘에 따라 마스터 프로세스와, 마스터와 동일하게 동작하는 슬레이브 프로세스로 구성된다. 여러대의 컴퓨터와 디스플레이로 구성된 대형 가시화시스템에서 동작할 경우, 서로 다른 GIVI 슬레이브 프로세스가 각 디스플레이의 렌더링을 담당하게 된다.

GIVI의 기능은 크게 3가지로 나눠볼 수 있는데, 그 기능은 각각 다음과 같다.

- GLOVE의 렌더링 엔진(GLORE)에 데이터를 요청하고, GLORE에서 가공되고 렌더링된 데이터를 전송받아서 출력한다. 이 기능은 GIVI 인터페이스에서 가장 기본적이면서도 가장 중요한 기능이라 할 수 있다.
- 개발자가 전체 화면이나 메뉴를 손쉽게 구성하는데 필요한 기반

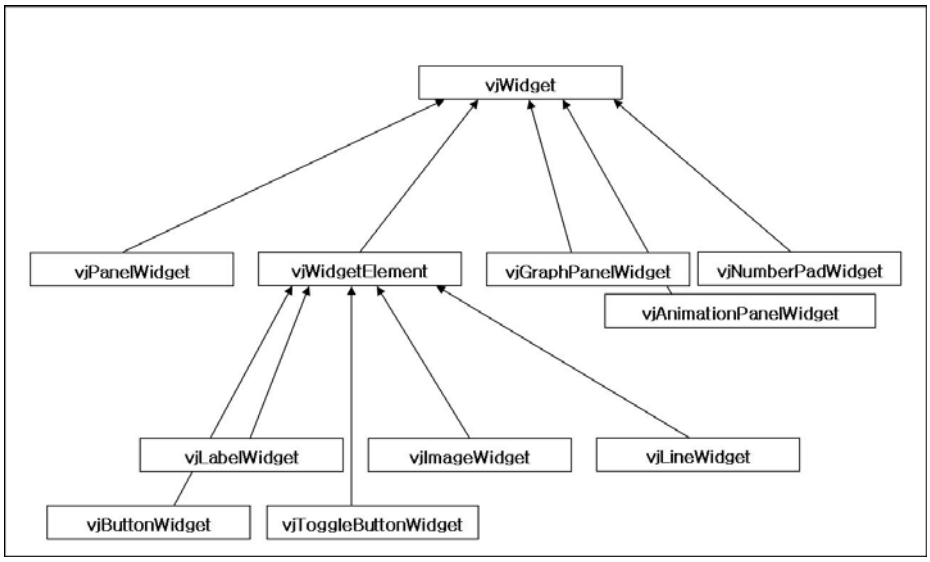
환경을 제공한다. GIVI는 vjVTK와의 연동을 통해 각종 2차원 위젯을 제공함으로써 사용자가 다양하게 화면을 구성할 수 있게 해준다.

- VRJuggler로부터 받아들이는 트래킹정보 및 이벤트 정보를 GLORE에 전달함으로써 가상현실 환경에서 사용자가 상호작용을 통해 가시화, 또는 시뮬레이션 과정을 제어할 수 있게 하면서 이와 관련된 콜백(Callback) 메커니즘을 제공한다. 즉, GIVI 인터페이스 환경을 통해 사용자의 동작에 따라 GLOVE의 동작을 제어하는 것이 가능하며, 이런 이벤트에 따른 동작을 구현하는 것도 가능하다. 특히, 특정 분야에 특화된 인터페이스를 구성하면 GLOVE 환경에 대한 사용자 편이성을 높이는 것이 가능하다. 현재는 로터 동역학 분야에 대한 인터페이스 환경을 구축하고 있다.

나. 클래스

여기에서는 GIVI의 위젯 클래스 구조에 대해 설명할 것이다. 위젯은 사전적으로는 사용자가 변경할 수 있는 정보를 배치해서 보여주는 GUI(Graphical User Interface)를 가리키는 용어인데, 가상 현실 환경에서는 사용자가 직접 3차원 공간에서 기하 정보를 이용해서 조작할 수 있는, 모든 형태의 인터페이스를 가리킨다.

GIVI의 위젯 클래스 구조는 [그림 4-2]와 같다.



[그림 4-2] GIVI 위젯 클래스 구조

다음은 각 클래스에 대한 설명이다.

1) viWidget

vjPanelWidget 및 패널 위에 올라가는 모든 위젯들의 상위에 위치하는 추상 클래스로, vjPanelWidget 및 패널 위에 올라가는 모든 위젯은 vjWidget 클래스를 상속받는다.

- 멤버 변수
  - mBorderElement: float
    - 패널 위에 위젯 구성요소들을 배치할 때, 구성요소간의 간격을 나타내는 상수로, 모든 위젯에 대해 공통값을 나타내며 기본값은 0.01
  - mPosition: float[3]
    - 위젯의 위치를 나타내는 변수
  - mSize: float[3]
    - 위젯의 크기를 나타내는 변수
  - mColor: float[3]
    - 위젯의 색상을 나타내는 변수
- 멤버 함수
  - SetPosition(value: float[3]): void
  - SetPosition(value: float, float, float): void
  - GetPosition(): float *
  - SetSize(alue: float[2]): void
  - SetSize(vlalue: float, float): void
  - GetSize(): float *
  - SetCOlor(value: float[3]): void
  - SetCOlor(value: float, float, float): void
  - GetColor(): float *

2) vjWidgetElement

vjPanelWidget 위에 초가할 수 있는 모든 widget element에 대한 추생 클래스로, vjPanelWidget을 제외한 모든 widget element들은 이 클래스를 상속한다.

- 멤버 변수
  - mRenderer: vrj::vjRenderer * : 렌더링에 대한 주체
- 메소드
  - SenRenderer(value: vrj::vjRenderer *): void

3) vjPanelWidget

vjPanelWidget은 widget element들을 올릴 수 있는 틀로, 메뉴 구성의 기본 요소라 할 수 있으며, vjWidget 클래스를 상속한다.

- 멤버 변수
  - mRenderer: vrj::vjRenderer *
  - mHeightElement: float
  - mTopXElement: float
    - mTop의 크기를 지정하는 상수
  - mIsVert: bool
    - 패널에 element가 세로, 혹은 가호 방향으로 추가되는지를 나타내는 값
    - true면 세로방향, false면 가로방향
  - mPanelName: char *
    - 패널상에 나타나는 텍스트를 나타낸다.
    - 메뉴의 제목이 될 수도 있다.
  - mNextVertPos: float
    - 다음에 widget element가 추가될 위치의 y좌표
    - 즉, 현재 y축 방향으로 패널의 끝을 나타내는 좌표값
  - mNextHoriPos: float
    - 다음에 widget element가 추가될 위치의 x좌표
    - 즉, 현재 x 축 방향으로 패널의 끝을 나타내는 좌표값

- mMenu: vtkPlaneSource *
  - 패널의 틀을 나타내는 기하 정보
- mTop: vtkPlaneSource *
  - 패널의 제목 부분에 위치하는 기하 정보
- mMenuMapper: vtkPolyDataMapper *
  - mMenu와 연동되는 poly data mapper
- mTopMapper: vtkPolyDataMapper *
  - mTopMenu와 연동되는 poly data mapper
- mMenuActor: vtkActor *
  - mMenu와 연동되는 액터
- mTopActor: vtkActor *
  - mTopMenu와 연동되는 액터
- mTopText: vtkVectorText *
  - 패널의 제목을 텍스트로 표현하는데 필요한 구성요소
- mTopTextMapper: vtkPolyDataMapper *
  - mTopText와 연동되는 poly data mapper
- mTopTextActor: vtkActor *
  - mTopText와 연동되는 액터

● 멤버 함수

- AddElement(value: vjWidgetElement *): void
  - widget element를 패널에 추가하는 루틴
  - 추가하는 widget element의 크기를 가지로 패널의 크기를 재설정하고, 다음에 추가될 위젯의 위치를 계산하는 CalculateNextPos 루틴을 호출한다.
- PanelInit(): void
  - 패널 위젯을 초기화하는 루틴으로 패널에 대한 size와 position을 설정한 후 호출해야 한다.
  - 패널에 대한 size와 position 값을 가지고 메뉴 패널의 크기

와 위치를 설정한 뒤, 패널 제목에 해당하는 부분의 크기와 위치를 설정해서 그리는 역할을 수행한다.

- SetMenuTop(): void
  - PanelInit 함수에 의해 호출되는 루틴
  - mSize와 mPosition, mBorderElement, mHeightElement 및 mTopXElement값을 이용해서 패널 제목에 해당되는 부분의 크기와 위치를 설정한다.
- SetTopText(): void
  - mSize와 mPosition을 이용해서 패널 제목에 해당하는 텍스트를 설정한다.
- SetPanelName(value: char *): void
  - 패널 제목에 해당하는 텍스트를 설정하는 루틴
- SetRenderer(value: vrj::vjRenderer *, vtkVJPicker *): void
  - 값으로 전달받은 인자에 패널 관련 액터들을 추가함으로써 렌더링 목록에 추가하는 루틴
  - mTopActor를 picker의 PickList에 추가함으로써 패널 제목 부분에 대해 picking 이벤트 등록이 가능하다.
- CalculateNextPos(value: float, float, float): void
  - 다음번 element가 추가될 위치를 계산해서 저장하는 루틴으로 mNextVertPos와 mNextHoriPos를 새로 계산해서 지정한다.
  - mIsVert값이 true인 패널에서는 mNextVertPos가 새로 계산되고, false인 패널에서는 mNextHoriPos가 새로 계산된다.
- SetPanelColor(value: float, float, float, float): void
  - 패널의 색상을 설정하는 루틴
  - 인자로 받아들이는 float 값은 각각 r,g,b,a를 나타낸다.
- SetTextColor(value: float, float, float, float): void
  - 패널제목을 나타내는 텍스트의 색상을 설정하는 루틴
  - 인자로 받아들이는 float 값은 각각 r,g,b,a를 나타낸다.

- SetIsVert(value: bool): void
  - 패널에 element로 추가되는 방향을 지정하는 속성으로 인자가 true면 세로 방향으로, false면 가로 방향으로 추가된다.
- GetPanelName(): char *
- GetNextPosition(): float *

4) vjButtonWidget

vjWidgetElement를 상속받은 위젯으로, 주로 픽킹(picking) 이벤트를 발생시켜서 다른 처리 결과를 도출하는 역할을 수행한다. 메뉴에서 결과 도출과정을 시작하는 이벤트를 발생시키는 버튼의 역할을 수행하는 위젯으로, 주로 vjPanelWidget 위에 올려서 메뉴를 구성하는 용도로 사용된다.

- 멤버 변수
  - mButtonName: char*
    - 버튼 위젯상의 텍스트를 나타내는 문자열
  - mRenderer: vrj::vjRenderer *
    - 버튼 위젯을 렌더링하는데 필요한 렌더러를 나타냄.
  - mActors: vtkActorCollection *
    - 버튼 위젯에 관련된 액터들, 즉 mButtonActor와 mNameActr를 모두 모아놓은 actor collection으로, 패널 위젯에서 버튼과 관련된 제어를 수행하는데 사용된다.
  - mButton: vtkCubeSource *
    - 버튼을 나타내는 기하정보
  - mButtonMapper: vtkPolyDataMapper *
    - mButton과 연동되는 poly data mapper
  - mButtonActor: vtkActor *
    - mButton과 연동되는 액터
  - mName: vtkVectorText *
    - 버튼 위젯의 제목을 텍스트로 표현하는데 필요한 구성요소

- mNameMapper: vtkPolyDataMapper *
  - mName과 연동되는 poly data mapper
- mNameActor: vtkActor *
  - mName과 연동되는 액터
- mButtonCB: vtkCallbackCommand
  - 버튼 위젯의 동작을 결정짓는 콜백 커맨드를 지정하는데 사용

● 멤버 함수

- SetName(value: char *): void
  - 버튼 위젯에 표시되는 텍스트를 설정하는 루틴
  - mName의 텍스트 속성을 설정
- SetPosition(value: float, float, float): void
  - mButtonActor와 mNameActor의 위치를 설정
  - mButtonActor는 중심점을 계산해서 설정하고, mNameActor는 x중심, y 아래 중심을 설정한다.
- SetPosition(value: float a[3]): void
  - SetPosition(float, float, float) 루틴을 호출한다.
- SetSize(value: float, float, float): void
  - mSize 값을 설정한 뒤, mButton의 XLength, YLength, ZLength 속성을 설정한다.
  - 이 때, ZLength는 상수값을 가진다.
- SetSize(value: float[3]): void
  - SetSize(float, float, float) 루틴을 호출한다.
- SetColor(float, float, float, float): void
  - 버튼 위젯의 색상을 설정하는 루틴이다.
  - 인자로 받아들인 float 값은 각각 r,g,b,a를 나타낸다.
- SetTextColor(float, float, float, float): void
  - 버튼 위젯에 표시되는 텍스트를 설정하는 루틴

- 인자로 받아들인 float 값은 각각 r,g,b,a를 나타낸다.

- SetCallback(value:void(*f)(vtkObject *caller, unsigned long eid, void *clientdata, void *calldata), void *clientdata)
  - 인자로 받아들인 함수포인터가 지정하는 함수를 버튼의 Pick 이벤트에 대한 콜백 함수로 설정한다.
  - 콜백 함수 f는 vtkCallbackCommand 클래스인 mButtonCB에 콜백함수로 등록되고, clientdata는 이 함수에 대한 client data로 설정된다.
  - mButtonCB는 AddObserver 함수를 통해 mButtonActor와 mNameActor에 콜백 함수로 추가됨으로써, 버튼에 대한 콜백이 설정된다.

- SetRenderer(value: vrj:vjRenderer *, vtkVJPicker *):void
  - 값으로 전달받은 인자(vrj::퍼Renderer *renderer)에 버튼 관련 actor를 추가함으로써 렌더링 목록에 추가하고, picking 리스트에 관련 액터들을 추가함으로써 버튼에 대한 pick 이벤트의 발생이 가능케 한다.
  - PickList에 추가되는 액터는 mButtonActor와 mNameActor 다.

5) vjLabelWidget

vjLabelWidget 클래스는 vjPanelWidget상에 텍스트 정보를 나타내는데 사용되며, 단독으로 사용해서 에러 메서지를 출력하거나 사용자에게 정보를 보여주는 용도로 사용할 수 있다.

vjLabelWidget에 대한 멤버 변수 및 멤버 함수는 그 명칭에서 직관적으로 파악할 수 있으므로 그 내용에 대해 따로 기술하지 않겠다.

- 멤버 변수
  - mRenderer: vrj::vjRenderer *
  - mLabel: vtkCubeSource *
  - mLabelMapper: vtkPolyDataMapper *
  - mLabelActor: vtkActor *

- mText: vtkVectorText *
- mTextMapper: vtkPolyDataMapper *
- mTextActor: vtkActor *

● 멤버 함수

- SetText(value: char *) : void
- GetText(): char *
- SetPosition(value: float, float, float): void
- SetPosition(value: float[3]): void
- SetSize(value: float, float, float): void
- SetSize(value: float[3]): void
- SetColor(value: float, float, float, float): void
- SetColor(value: float[4]): void
- SetTextColor(value: float, float, float, float): void
- SetTextColor(value: float[4]): void
- SetTextScale(value: float, float, float): void
- SetTextScale(value: float[3]): void
- SetRenderer(value: vrj::vjRenderer *renderer): void

6) vjLineWidget

vjLineWIdget은 vjWidgetElement 클래스를 상속받아서 패널 위에서 사용할 수 있으며, 주로 항목간 구분자를 넣는 용도로 사용된다. vjLinelWidget에 대한 멤버 변수 및 멤버 함수는 그 명칭에서 직관적으로 파악할 수 있으므로 그 내용에 대해 따로 기술하지 않겠다.

● 멤버 변수

- mLine: vtkLineSource *
- mLineMapper: vtkPolyDataMapper *
- mLineActor: vtkActor *

● 멤버 함수

- PlaceWidget(): void

- SetRenderer(value: vrj::vjRenderer *): void
- SetIsVert(value: bool): void
- SetPosition(value: float, float, float): void
- SetPosition(value: float[3]): void
- SetSize(value: float, float, float): void
- SetSize(value: float[3]): void
- SetLength(value: float): void
- SetColor(value: float, float, float, float): void
- SetColor(value: float[4]): void

7) vjImageWidget

vjImageWidget은 vtkImageData를 패널 위에서 디스플레이하기 위한 클래스로 vjWidgetElement를 상속한다. vjImageWidget의 멤버 변수 및 멤버 함수는 그 명칭에서 직관적으로 파악할 수 있으므로, 그 내용에 대해 따로 기술하지 않겠다.

- 멤버 변수
  - mPanel: vtkPlaneSource *
  - mPanelMapper: vtkPolyDataMapper *
  - mPanelActor: vtkActor *
  - mTexture: vtkTexture *
  - mData: vtkImageData *
- 멤버 함수
  - SetData(value: vtkImageData *): void
  - SetPosition(value: float, float, float): void
  - SetPosition(value: float[3]): void
  - SetSize(value: float, float, float): void
  - SetSize(value: float[3]): void
  - SetRenderer(value: vrj::vjRenderer *): void
  - GetData(): vtkImageData *

8) vjToggleButtonWidget

vjToggleButtonWidget은 vjWidgetElement를 상속받으며, 패널 위젯 위에서 사용되는 토글 버튼을 구현할 목적으로 설계됐다. 이 위젯은 파라미터에 대한 on/off 상태를 나타내는데 사용된다. vjToggleButton Widget의 멤버변수 및 멤버 함수는 그 명칭에서 직관적으로 파악할 수 있으므로, 그 내용에 대해 따로 기술하지 않겠다.

- 멤버 변수
  - mButtonName: char *
  - mOnOff: bool
  - mToggleCB: vtkCallbackCommand *
  - mTButton: vtkCubeSource *
  - mTButtonMapper: vtkPolyDataMapper *
  - mTButtonActor: vtkActor *
  - mTSphere: vtkSphereSource *
  - mTSphereMapper: vtkPolyDataMapper *
  - mTSphereActor: vtkActor *
  - mTName: vtkVectorText *
  - mTNameMapper: vtkPolyDataMapper *
  - mTNameActor: vtkActor *
- 멤버 함수
  - SetName(value: char *): void
  - SetPosition(value: float, float, float): void
  - SetPosition(value: float[3]): void
  - SetSize(value: float, float, float): void
  - SetSize(value: float[3]): void
  - SetColor(value: float, float, float, float): void
  - SetColor(value: float[4]): void
  - SetToggleStatus(value: bool): void

- GetToggleStatus(): bool
- SetRenderer(value: vrj::vjRenderer, vtkVJPicker *): void

9) vjGraphPanelWidget

vjGraphPanelWidget은 GLOVE의 그래프 생성 결과를 보여주는데 사용되는 위젯으로 vjWidget을 상속한다.

10) vjNumberPadWidget

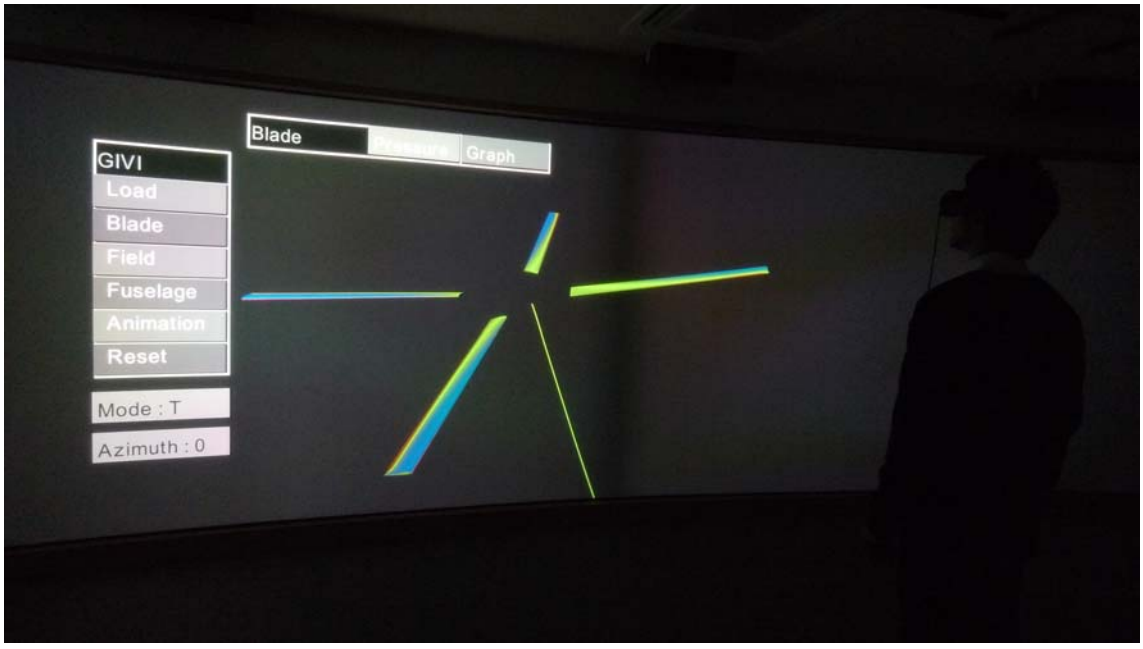
vjNumberPadWidget은 따로 키보드 입력이 없는 가상현실 환경에서 숫자 입력을 목적으로 설계된 위젯으로, vjWidget 클래스를 상속한다.

11) vjAnimationPanelWidget

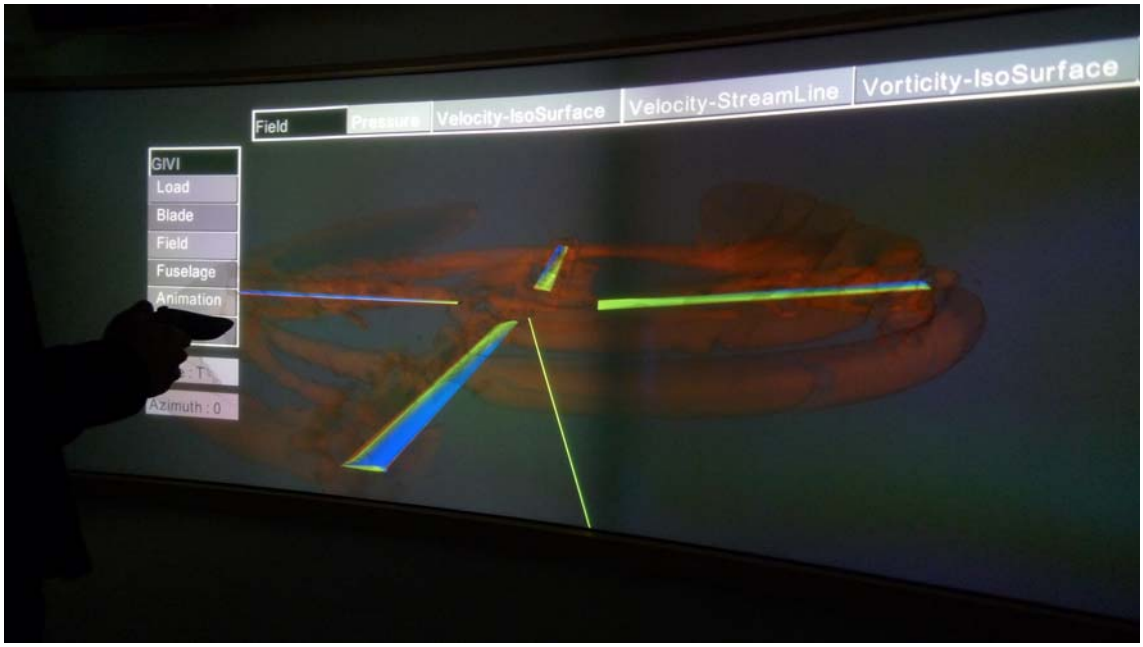
vjAnimationPanelWidget은 steady 데이터가 아닌 time-varying 데이터를 제어할 때, time-step 별로 데이터를 렌더링하는 목적으로 설계된 위젯이다.

5. GIVI의 인터페이스 환경

GIVI 인터페이스를 사용해서 구성한 GLOVE 실행 환경은 [그림 5-1]과 같다.



[그림 5-1] GLOVE 실행 환경 (블레이드 데이터)



[그림 5-2] GLOVE 실행 환경 (필드 데이터)

[그림 5-1]과 [그림 5-2]는 GLOVE에서 로터 데이터를 가시화한 사례로, 왼편과 상단의 메뉴를 이용해서 블레이드 데이터를 가시화할지, 혹은 필드 데이터를 가시화할지를 선택할 수 있다. [그림 5-1]은 블레이드 데이터를 가시화한 것으로, 블레이드 데이터에서는 그래프를 생성하는 것이 가능하다. 반면 [그림 5-2]는 필드 데이터를 가시화한 것으로, 필드 데이터에서는 각종 파라미터를 조절해서 가시화 수위를 조절할 수 있다. 또, 사용자는 wand같은 가상현실 기기를 이용해서 오브젝트에 대해 transformation, rotation, zooming 등의 동작을 수행하는 것도 가능하다.

블레이드 데이터에 대해 가시화할 수 있는 항목은 다음과 같다.

- Pressure distribution on blades: 블레이드 표면의 압력 분포를 보여준다.
- 데이터에 대한 정량적 분석
  - Pressure distribution: 주어진 span position에서 회전각에 따른 압력의 변화를 정량적으로 보여준다.
  - Pitch angle variation: 특정 span position과 0~360도의 회전각에 대한 pitch angle의 변화를 보여준다.
  - Sectional force: 주어진 span position과 회전각에 대한 section

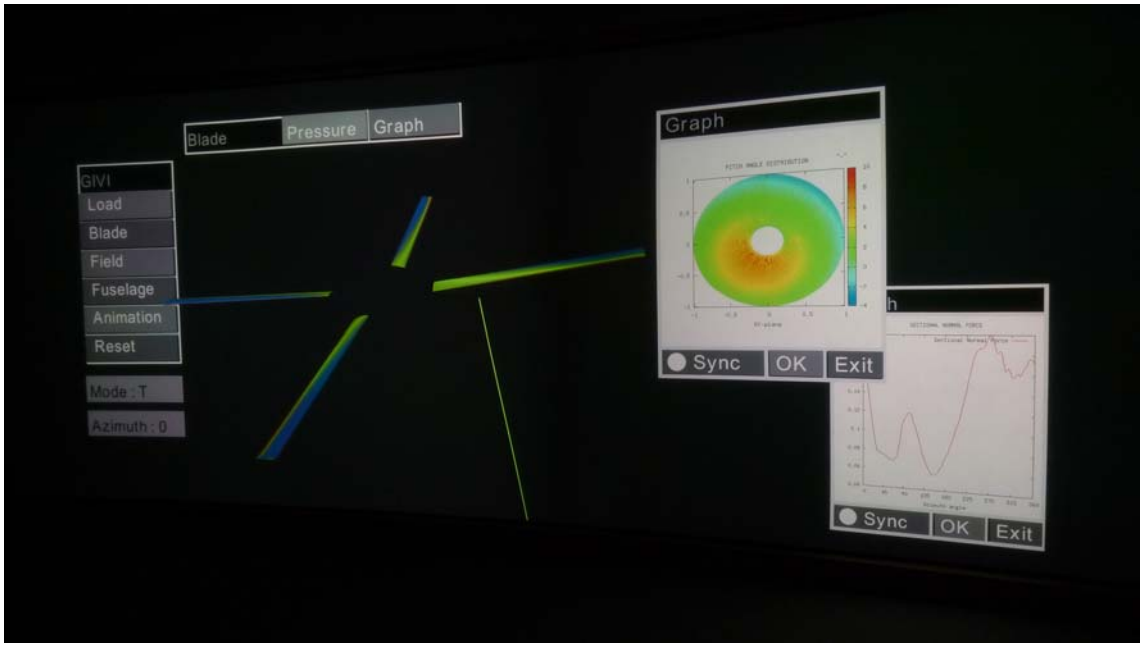
al normal force, sectional chord force, sectional span force 정보를 보여준다.

- Sectional moment: span position과 회전각에 대한 sectional x-moment, sectional y-moment, sectional z-moment 그래프를 커브와 디스크 형태로 제공한다.

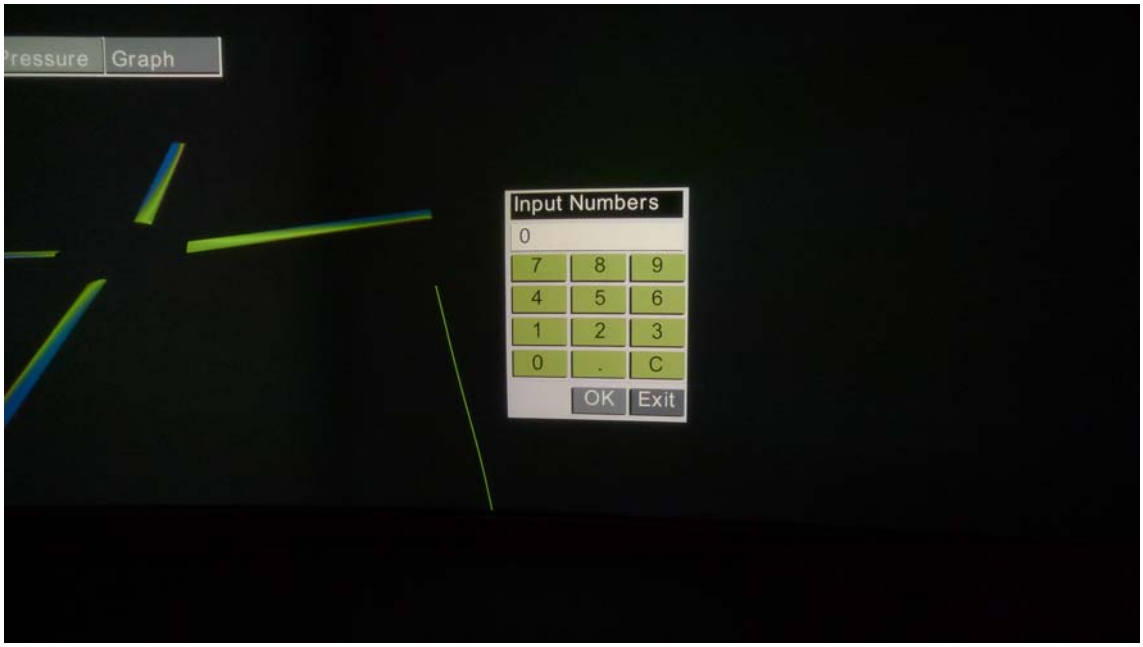
- Animation: 시간의 흐름에 따른 블레이드 표면의 압력분포 변화를 그래프의 타임스텝과 동기화해서 애니메이션으로 보여준다.

사용자는 필드 데이터 가시화를 통해 블레이드와 필드 데이터 사이의 상호작용을 명확하게 시각적으로 확인할 수 있다. GLOVE에서 필드 데이터에 대해 가시화할 수 있는 항목은 다음과 같다.

- Vorticity iso-surface: 주어진 vorticity 값을 갖는 iso-surface를 보여준다. 보고자 하는 vorticity의 크기값을 입력받아 그릴 수 있다.
- Q-Criteria: vortex 영역을 Q-criteria 기법을 사용하여 보여준다.
- Velocity streamline: 관심있는 영역을 지정하거나 vorticity 값이 특정값 이상인 부분에서 velocity streamline을 보여줄 수 있다.
- Velocity iso-surface: 사용자가 지정한 velocity 값을 갖는 iso-surface를 보여준다. 사용자가 그리고자 하는 velocity 크기값을 지정할 수 있다.



[그림 5-3] 그래프 패널



[그림 5-4] vjNumberPadWidget

- Animation: 블레이드 데이터와 vorticity, velocity에 대한 데이터를 가시화하여 블레이드 회전시 vorticity와 velocity 변화를 관찰할 수 있다.

[그림 5-3]과 [그림 5-4]는 각각 vjImageWidget을 통해 그래프를 보여주는 vjGraphPanelWidget과 vjNumberPadWidget을 나타낸다. 사용자는 GLOVE 환경에서 데이터와 함께 그래프를 확인해볼 수도 있고, vjNumberPadWidget을 통한 숫자 입력을 통해 자신이 원하는 값을 정확하게 지정해서 가시화해볼 수 있다.

6. 결론

GLOVE는 가상현실 환경 내에서 복잡한 대용량 데이터를 실시간으로 가시화하고 그 결과를 원격지 사용자와 공유할 수 있는 통합 환경을 제공한다. 이와 유사한 목적으로 개발된 여러 가시화 도구들이 범용 사용자 환경을 제공하는 것과는 달리, 응용에 따라 맞춤형 사용자 환경을 제공하는 것을 목표로 하며, 이를 위해 GLOVE는 어떤 분야의 데이터라 할지라도 가시화할 수 있게 하기 위한 계층적 데이터 관리 구조를 가지며, 대용량 데이터를 효율적으로 처리하는 구조를 가진다. 또, 고해상도의 타일형 디스플레이와 가상현실 환경을 제공함으로써

응용 데이터의 해석을 용이하게 할 수 있게 한다.

GLOVE의 GIVI는 GLOVE 프레임워크에서 사용자와 관련된 제반 인터페이스를 제공하는 프레임워크로, 렌더링 엔진으로부터 렌더링된 결과를 받아서 출력하는 것 뿐만 아니라 GLOVE 환경에서 사용자가 상호작용을 통해 데이터를 제어하는데 필요한 모든 기반환경을 제공하는데 그 목적을 둔다. 현재는 로터 동역학 분야에 대한 맞춤형 사용자 인터페이스를 개발함으로써 사용자 편이성을 높이는 것을 그 목표로 두고 있다. 향후에는 GLOVE 의 대용량 데이터 처리를 위한 병렬 입출력과 분산 데이터 관리, 병렬 렌더링 기능을 추가로 구현하면서 GIVI의 이 부분에 대한 인터페이스 부분 개발이 주력할 예정이며, 특히 로터 동역학 분야의 데이터 분석에 필요한 3차원 인터페이스 개발에 주력하게 될 것이다.