ISBN 978-89-6211-277-1

가시화 소프트웨어에 대한 연동성을 지원하는 Virtual Reality 시스템 구축에 관한 연구

(Study on the Development of Virtual Reality System to Support 2-D Based Visualization Softwares)

금 복 희 (Bokhee Keum)
bhkeum@kisti.re.kr
Supercomputing Infrastructure Team, Supercomputing Center

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

제목 차례

표 차례

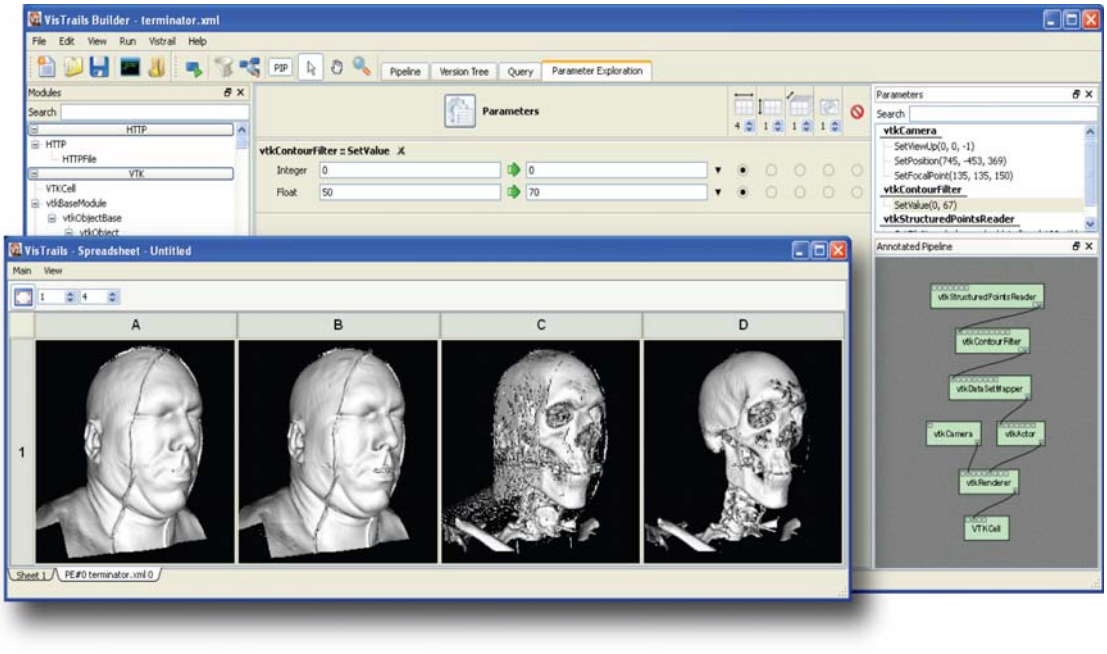
그림 차례

1. 배경

Computer simulation은 물리, 화학, 생물, 기계공학 등 자연과학/공학뿐만 아니라 인문/사회과학 분야에서도 많이 적용되고 있는 연구방법이다. Computer simulation은 computation과 visualization이라는 두 가지 기술에 의해 발전되고 있는데 고성능 컴퓨터(HPC : High Performance Computer)를 포함한 Computation 기술의 빠른 발전은 기존에 적용하기 어려웠던 문제를 빠른 시간 내에 해결하는 것을 가능하게 하였으며 새로운 현상의 발견도 가능하게 함으로써 컴퓨터를 이용한 시뮬레이션이 가속화되고 있다.

시뮬레이션 결과는 시뮬레이션 소프트웨어에서 제공하는 파일포맷으로 바이너리 형태 혹은 텍스트 형태로 저장이 되는데 연구자가 이 파일의 내용을 그대로 보면서 결과를 해석하는 것은 대부분 불가능한 일이다. 시뮬레이션 결과를 연구자가 쉽게 알아볼 수 있도록 시뮬레이션 데이터에 그래픽처리를 하여 보여주는 기술이 scientific visualization이다. 그러나 기하급수적으로 발전하는 computation 기술에 비하여 visualization 기술의 발전은 이를 따라가지 못하고 있으며 simulation scientist가 graphic workstation이나 PC를 이용한 기존의 방식으로 대용량의 복잡한 결과 데이터를 이해하기가 어려워지고 있다. 이에 사용자들은 2d display에서의 단순한 interface 보다 더 발전된 방식을 요구하고 있으며 이러한 사용자 요구에 대한 해결방법으로 VR (Virtual Reality, 가상현실. 이하 VR) 기술이 적용되고 있으며 VR 적용 시 장점이 많이 보고되고 있다[5,6]. 그러나 VR system은 개발에 소요되는 비용과 노력이 2d visualization system의 개발에 비해 상당히 크며 개발된 대부분의 VR 시스템은 기존의 visualization system과 완전히 독립적으로 구성이 되어 있어서 기존의 visualization system과의 자연스러운 연동기능이 제공되지 않거나 특정한 system에 밀접하게 구성되어 있어 VR system의 활용성이 떨어진다.

또한 simulation 결과의 정확한 해석을 위해서는 동일한 데이터에 대해서 다양한 visualization 기능을 적용하거나 유관 데이터에 대해서 각각 visualization을 적용하여 한 화면에 동시에 그 visualization 결과를 display하는 Multi-View 기능([그림 1] 참조)이 visualization system의 중요한 기능으로 부각되고 있으나 기존의 VR 시스템에서는 이러한 기능을 제공하고 있지 않다.



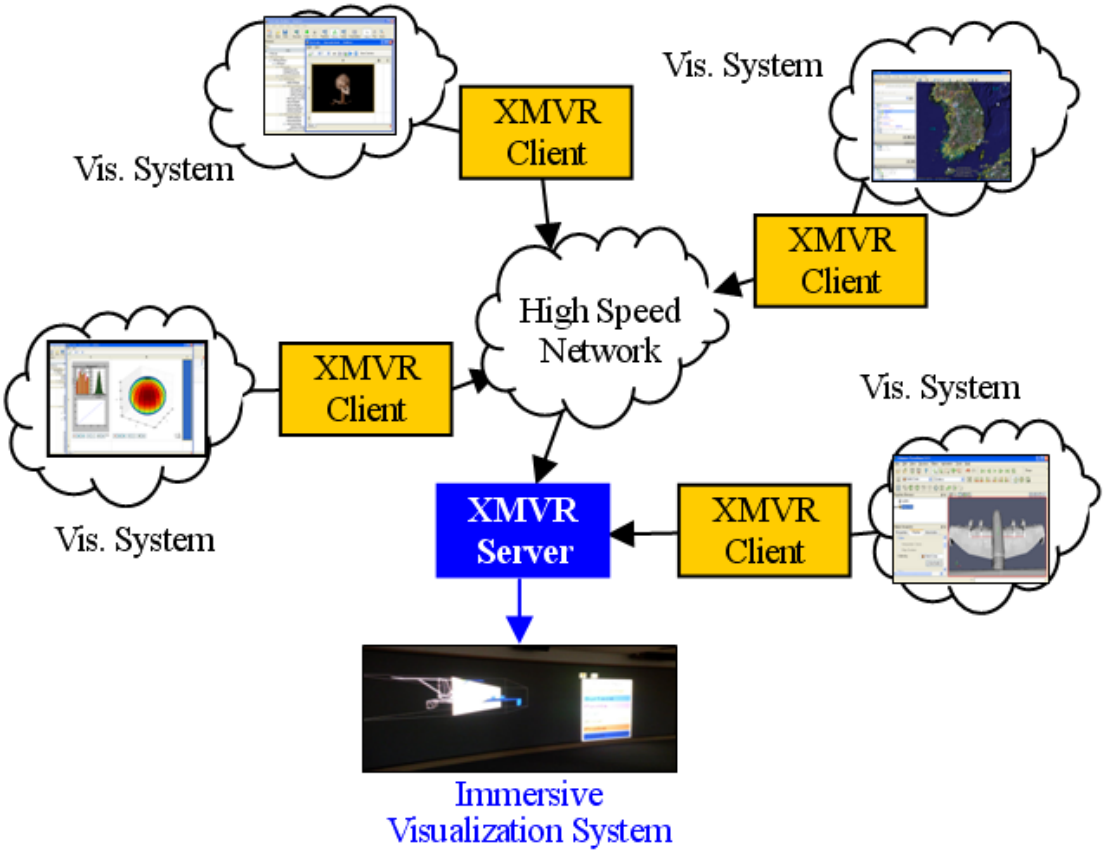
[그림 2] VisTrails의 Multiview 예
(http://www.vistrails.org/index.php/Documentation에서 발췌)

따라서 본 보고서에서는 기존의 visualization system에 VR 기능을 쉽게 적용할 수 있고 확장성이 뛰어난 framework를 제안하고 simulation 결과에 대한 정확한 해석에 도움이 될 수 있도록 2개 이상의 visualization 결과를 동시에 다룰 수 있는 Multi-View VR system의 구현방법을 설명하고자 한다.

제안하는 framework의 이름은 XMVR (eXtensible Multiview Virtual Reality)이며 XMVR framework의 architecture와 구체적인 구현 방법, 그리고 Immersive Display에서의 Multiview 처리방법의 순서로 내용이 구성되어 있다.

2. XMVR Framework의 Architecture

XMVR은 remote VR Client인 visualization software가 네트웍으로 가시화 데이터 혹은 이 소프트웨어로 얻은 가시화 결과를 몰입환경에서 interaction이 가능하게 하는 VR Server로 전송하면 VR Server는 자동으로 immersive display를 실행한다. 따라서 VR Client site에서는 추가의 비용을 들여서 VR system을 구축하지 않고도 VR Server에서 제공하는 VR 기능을 모두 활용할 수 있고 High Performance Rendering Cluster와 Visualization Cluster, 그리고 고해상도를 가지는 Projector로 구성된 VR system의 utilization을 높일 수 있는 장점을 가지고 있다.



[그림 3] XMVR Framework Architecture

가. VR Client

VR Client는 visualization software와 State Manager로 구성된다. ParaView와

같은 기존의 가시화용 소프트웨어가 visualization software에 해당하며 State Manager는 가시화 소프트웨어가 생성한 가시화 결과를 저장하는 State Saver와 이를 VR Server에 네트웍으로 전송하는 State Messenger로 이루어져 있다. 즉, 컴퓨터 시뮬레이션 결과는 Visualization Software를 이용하여 가시화 처리가 이루어지며 사용자는 VR로 결과를 display하고자 할 때 State Manager를 호출하면 State Manager는 가시화 결과를 그대로 저장한 후 시뮬레이션 데이터와 함께 VR Server의 State Loader로 전송한다. State Saver와 State Messenger의 역할과 특징을 정리하면 다음과 같다.

State Saver

1. Role : visualization software의 visualization 결과 저장
2. visualization software dependent
3. Module Type: Add-on module or dynamic plug-ins
4. Calls State Messenger

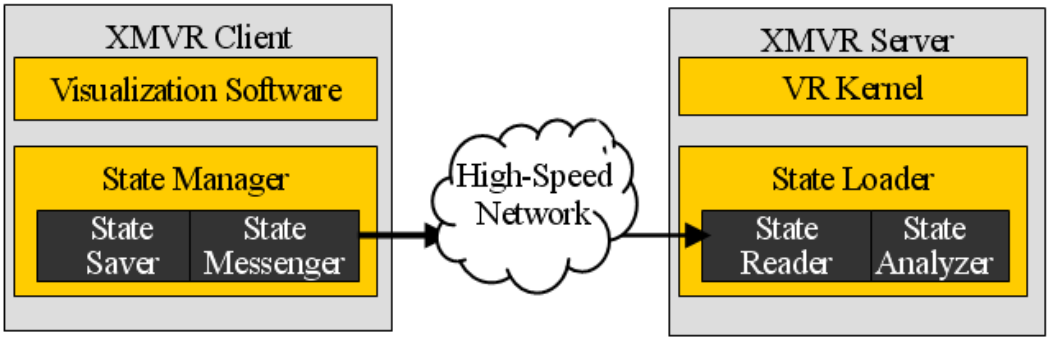
State Messenger

1. Role : 저장된 visualization 결과를 VR Server에 전송
2. visualization software independent
3. Module Type : library
4. Client process to State Reader of the VR Server

나. VR Server

XMVR의 VR Server는 State Loader와 VR Kernel로 구성이 되며, State Loader는 State Messenger에서 보내온 데이터를 수신하여 저장하는 State Reader와 visualization state file을 해석하여 working memory에 state 값을 설정하고 이를 VR 커널에서 사용할 수 있도록 하는 State Analyzer로 이루어져 있다. State Reader는 disk에 데이터를 저장한 후 State Analyzer에 VR service를 요청한다. State Analyzer는 visualization state 정보와 VR의 hardware/software에 맞게 VR service 수행환경을 설정하여 VR service 실행 준비를 한다. State Reader, Stat

e Analyzer의 역할과 특징을 정리하면 다음과 같다.



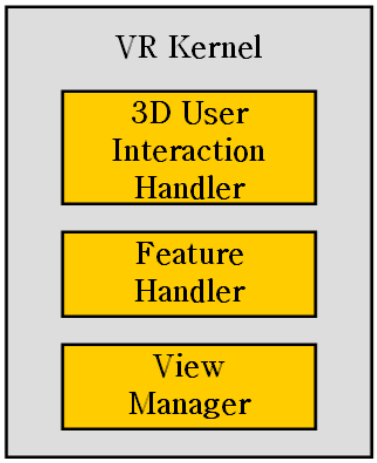
[그림 4] XMVR Client와 Server의 Network Communication 계층 구조

State Reader

1. Role : VR Client로부터 전송된 데이터를 저장하고 VR Service 요청
2. Module Type : stand-alone execution program
3. Server to the State Messenger of the VR Client

State Analyzer

1. Role : State Reader로부터 VR Service 요청을 접수하고 visualization state 파일을 해석하여 VR Kernel의 상태를 설정하고 VR Service 시작
2. Module Type : plain module tightly coupled with VR Kernel
3. Calls VR kernel modules



[그림 5] VR Kernel 구성 요소

	State Saver	State Messenger
Role	Storing the visualization result from the visualization systems	Transmission of the result information and the source data
System Dependency	Visualization System dependent	Visualization system independent
Module Type	Add-on or plug-in	API library or stand-alone program
Calling process	Calls State Messenger	Requests uploading service to State Reader of the VR Server

[표 1] State Saver와 State Messenger

VR Kernel은 그림[4]와 같이 크게 3D User Interaction Handler, Feature Handler, View Manager로 구성되어 있다. 3D User Interaction Handler는 6DOF(Degree Of Freedom)를 가지는 tracker과 wand의 버튼, 움직임 등을 처리한다. Feature Handler는 데이터에 적용되는 다양한 가시화 기능을 사용자의 입력에 따라 처리하는 역할을 담당한다.

View Manager는 VR space에 display되는 여러 widget을 처리함과 동시에 여러 개의 visualization 결과를 하나의 space에 display하는 방법과 User Interaction을 처리하는 일을 한다.

3. XMVR Framework의 구현

가. VR Client의 State Saver 구현

Visualization Software의 가시화 결과는 XML format의 파일로 저장이 되며 이 때 저장되는 주요 정보는 가시화 데이터의 파일이름과 위치, 적용된 visualization filter(operation), 조명, 칼라맵 등이 있다. 상태정보는 ParaView의 상태정보와 동일한 형식을 적용하고 하나의 가시화 결과는 하나의 파일에 저장이 된다. Multiview를 지원하는 software의 경우 하나의 가시화 결과에 여러 개의 view가 포함될 수 있으므로 multiview도 하나의 결과파일에 모두 저장된다.

상태파일의 예를 들면, 가시화 결과 중에서 데이터 파일과 조명에 대한 정보는 다음과 같은 형태로 저장된다.

```
<Proxy group="sources" type="LegacyVTKFileReader" id="78" servers="1">
  <Property  FileNames" id="78.FileNames" number_of_elements="1">
    <Element index="0" value="/xtmp/pvsm/Data/blow.vtk"/>
    <Domain  files" id="78.FileNames.files"/>
  </Property>
  <Property  TimestepValues" id="78.TimestepValues"/>
  <SubProxy  Reader" servers="1"/>
</Proxy>
```

[그림 6] 가시화 상태파일의 데이터 파일정보

가시화 대상이 되는 데이터 파일은 "/xtmp/pvsm/Data/blow.vtk"이고 이 파일의 형태는 Legacy VTK이다.

```
<Property  KeyLightIntensity" id="21.KeyLightIntensity" number_of_elements="1">
  <Element index="0" value="0.75"/>
  <Domain  range" id="21.KeyLightIntensity.range">
    <Min index="0" value="0"/>
    <Max index="0" value="2"/>
    <Resolution index="0" value="0.05"/>
  </Domain>
</Property>

<Property  LightAmbientColor" id="21.LightAmbientColor" number_of_elements="3">
  <Element index="0" value="1"/>
  <Element index="1" value="1"/>
  <Element index="2" value="1"/>
  <Domain  range" id="21.LightAmbientColor.range">
    <Min index="0" value="0"/>
    <Min index="1" value="0"/>
    <Min index="2" value="0"/>
    <Max index="0" value="1"/>
    <Max index="1" value="1"/>
    <Max index="2" value="1"/>
  </Domain>
</Property>
```

[그림 7] 가시화 상태파일의 조명정보

위의 설정은 가시화 결과를 display할 경우 조명에 대한 정보를 표현하고 있는데, scene의 주 조명이 되는 KeyLight에 대한 정보로 밝기값(0.75)을 나타내고, 주변광의 색상(1, 1, 1)에 대한 내용이 있다.

지원하고자 하는 visualization software에 대하여 이와 같은 형태로 상태정보를 저장하는 module을 C/C++로 작성하였으며 software에 plugin 형태로 추가할 수 있다.

나. VR Client의 State Messenger 구현

State Messenger의 주 역할은 앞에서 언급한 바와 같이 저장된 visualization state 파일과 data 파일을 VR server로 전달하는 것이다. 이를 구현하기 위해 QUANA[16]를 사용하였다. QUANTA는 과학자들의 협업 환경(Collaboration Environment)을 지원하는 목적으로 대량의 데이터를 고속으로 전송하기 위한 infrastr

ucture에 사용하기 위해 개발된 toolkit으로 다양한 platform에서 사용이 가능하며 사용자가 전송하고자 하는 데이터의 특성을 응용프로그램 수준에서 설정하면 QUANTA는 실제 네트웍 특성에 맞게 사용자의 설정을 변환하여 데이터를 전송하는 장점이 있으므로 remote VR Service 실행이 지원되어야 하는 XMVR framework에 적합한 툴이다.

QUANTA Client로 동작하는 State Messenger가 QUANTA Server인 State Reader에 파일 uploading을 요청하면 State Messenger가 데이터 파일과 visualization 결과 파일을 VR system의 local disk에 저장한다.

다. VR Server의 VR Hardware 구성

VR display 장치는 4개의 rear projector(SONY SRX-S110)를 가지는 Wall Screen으로 구성되어 있으며(표[1]) 입력장치는 FlyBox[20] joystick과 IS-900 MicroTrax[21]의 head tracker와 wand를 사용하고 있다.

Type of projector	SONY SRX-S110
Resolution of a single projector	4096 x 2160
Number of projectors	4
Stereoscopic image	INFITEC stereo
Computing system	Picasso
Installation year	2008

[표 2] Wall Screen H/W Specification

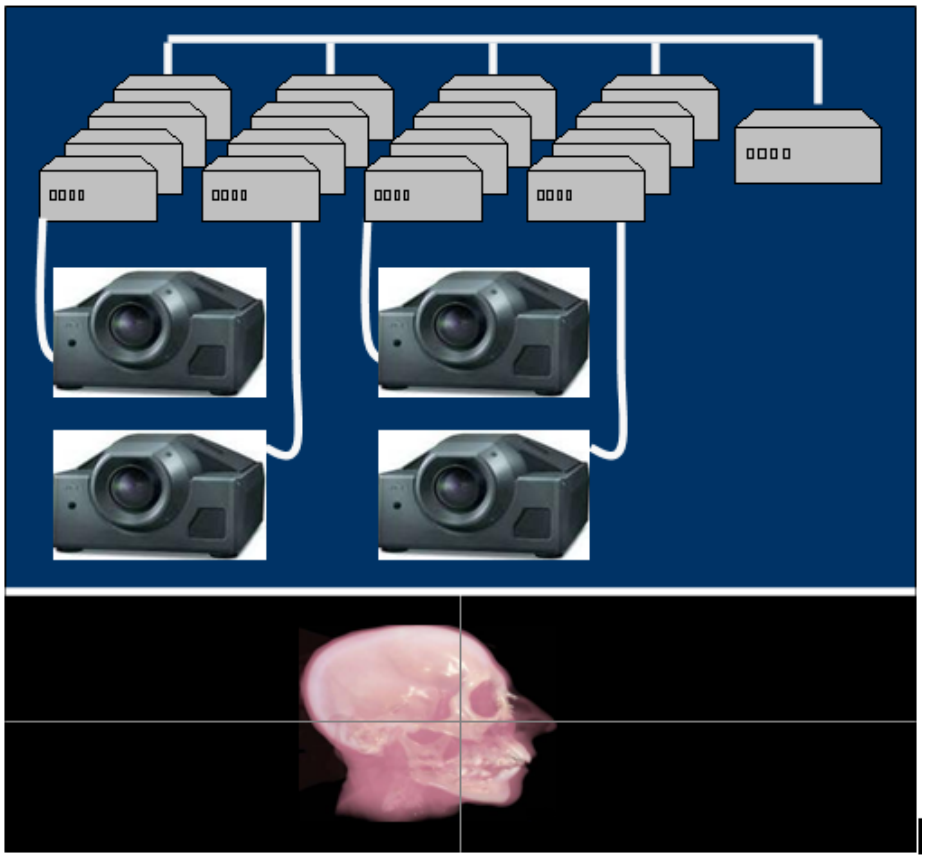
VR 기능 실행을 위한 Application Master Node의 사양은 표[2]와 같으며 immersive display용 node의 사양은 표[3]과 같다.

Node	Manufacturer	HP
	Model	xw8600
	# of nodes	2 + 1 spare
CPU	Manufacturer	Intel
	Model	Xeon X5450
	Clock	3.0 GHz
	# of FP operations / clock / core	4
	# of CPU cores / socket	4
	# of CPU sockets / node	2
	GFLOPS / core	12 GFLOPS
	GFLOPS / socket	48 GFLOPS
	GFLOPS / node	96 GFLOPS
Main memory		64GB
GPU	Manufacturer	NVIDIA
	Model	QuadroFX5600
	Graphic memory	1.5 GB
	Memory bandwidth	76.8 GB/sec
	Frame lock support	Yes
	Trangles / sec	300 million
	Texels / sec	38.4 billion

[표 3] VR System Maseter Node

Node	Manufacturer	HP
	Model	xw8600
	# of nodes	16
CPU		Master와 동일
Main memory		64GB
GPU		Master와 동일

[표 4] VR Display Node



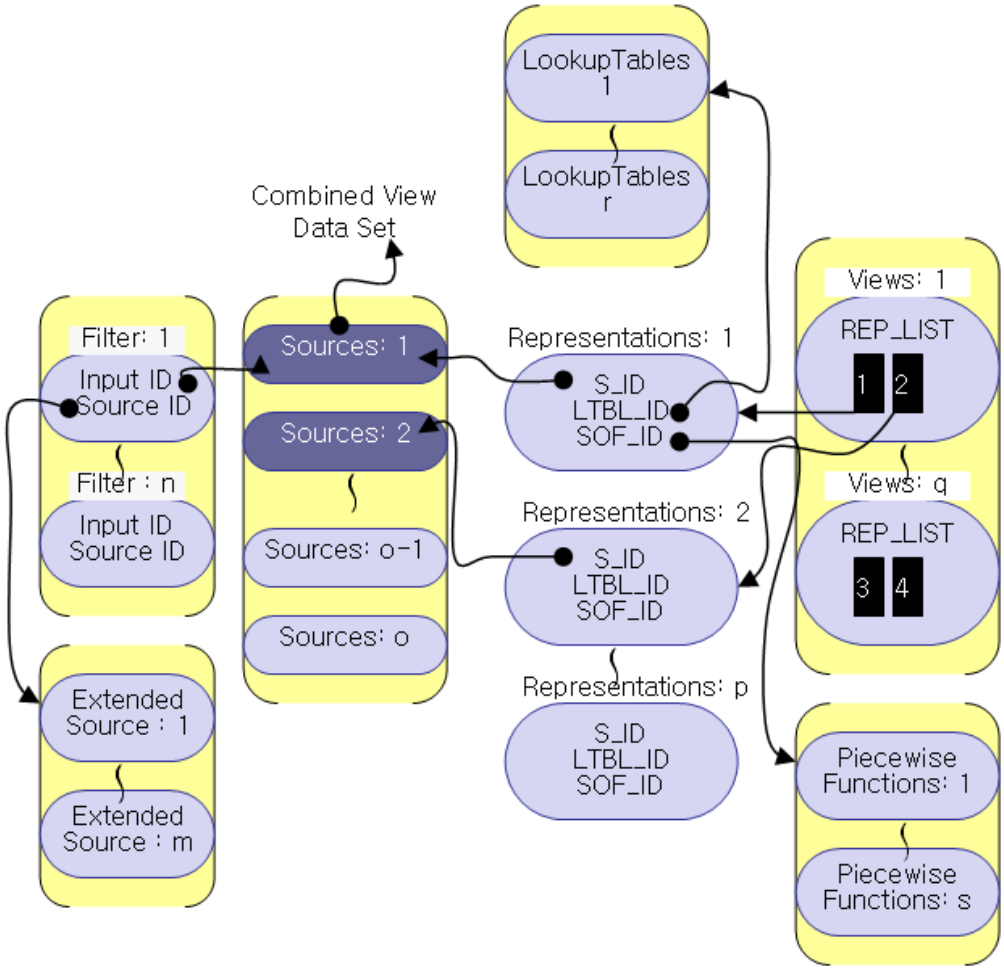
[그림 8] VR System Display 구성도

라. VR Server의 State Reader 구현

State Reader는 State Messenger와 같이 QUANTA로 구현했으며 QUANTA의 Server 프로세스로 State Messenger로부터 요청된 파일에 대해서 uploading service를 처리한 후 'system()' call로 VR service를 실행한다.

마. VR Server의 State Analyzer 구현

State Analyzer는 VR service를 실행하기 위하여 visualization 결과 파일을 해석하여 VR Kernel의 실행 상태를 설정한다.



[그림 9] 가시화 결과와 해석 구조

State Analyzer가 VR kernel로 로딩하는 visualization 정보의 종류는 'sources', 'representations', 'views', 'lookup_tables', 'piecewisefunctions', 'filters', 'extended_sources'이다.

- Sources

visualization의 대상이 되는 파일의 종류와 위치를 나타내는데 사용자가 visualization software에서 loading한 모든 데이터 파일은 각각의 'Sources' 필드를 가진다. 따라서 'Sources'의 빈도수와 동일한 개수의 data set에 초기 로딩 단계에서 VR Kernel에 전달된다.

- Representations

이 필드는 데이터나 데이터에 대한 visualization 처리 결과를 Rendering View에 display하기 위한 그래픽 정보를 유지하는 필드로 이 representations의 대상이 되는 'sources' 필드에 대한 ID와 함께 데이터의 geometry에 대한 색상, representation(points, wireframe, surface), ambient/diffuse/specular(주변광/난반사광/정반사광)에 대한 적용률과 색상 등에 대한 정보를 가지고 있다.

- Views

visualization software에서 사용한 rendering view에 대해 정보를 나타내는 element로 view의 크기, 위치, 배경색, 카메라/조명 정보, 그리고 이 view에 해당하는 데이터 파일의 'representations' element의 ID 등에 대한 정보를 유지하는 필드이다. 하나의 view에는 여러 데이터 파일이 로딩될 수 있으므로 다수의 'representations' ID가 하나의 view에 존재할 수 있다.

- Filters

visualization software에서 데이터 파일에 적용한 visualization filter에 대한 정보를 유지하는 필드로 Clipping, StreamTracer와 같은 filter들이 여기에 해당한다. filter 적용 대상은 'Input' ID로 나타낸다.

- Lookup_tables

display 시 사용하는 color space(RGB, HSV)를 정의하는 필드로 'representations' 필드에서 사용하고자 하는 'lookup_tables'에 대한 ID를 가지고 있다.

- Extended_sources

visualization 처리 결과로 생성된 visualization object로 filter 적용시 자동으로 생성된 object의 정보를 유지하는 필드로 StreamTracer를 실행할 경우 seed point가 여기에 해당한다.

State Analyzer가 visualization 데이터와 결과 정보를 loading하는 과정은 다음과 같다.

Step 1: state file로부터 각 필드('sources', 'representations', 'views', 'lookup_tables', 'piecewisefunctions', 'filters', 'extended_sources')의 정보를 추출하여 각

각의 리스트에 저장한다.

Step 2: 'sources' 필드의 data file을 open하고 data set의 종류가 Geometric data인지 Volumetric(image data 포함) data인지를 구별하고 해당 data set을 메모리로 로딩한다.

Step 3: multi-view 구성을 위하여 'views' 리스트에 있는 각 view node에 대해 representation list([그림8] 'Views: 1'의 REP_LIST 참조) 필드를 참조하여 이 view에 해당하는 representations node를 알아낸 후 이 node의 source id(S_ID), lookup_table id(LTBL_ID) 필드의 값으로 source data set의 node와 lookup table의 node reference 정보를 얻는다.

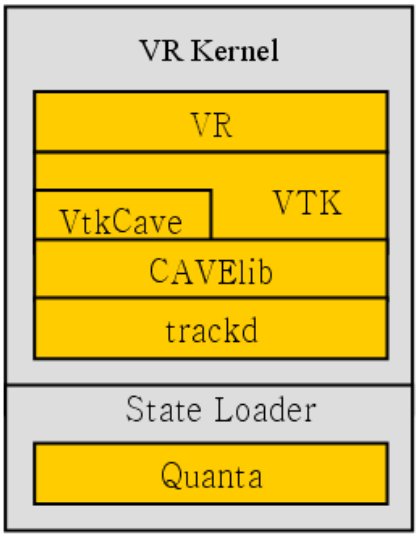
Step 4: Step 3의 reference 정보를 이용하여 각 view의 display object를 생성한다. 즉, source data set을 그래픽 object로 만들고 색상, 조명 등 representation을 정보를 설정한다.

Step 5: visualization software에서 사용한 filter를 VR에서 적용하기 위하여 'filters' 리스트를 탐색하면서 각 filter에 해당하는 'sources' node를 찾아, 그 node의 data set에 대해 filter를 적용한다. 이 때 'filters' node에 'extended_source'에 대한 ID를 가지고 있는 경우 filter 적용에 앞서 extended_source의 내용을 filter에 반영한다.

바. VR Server : Kernel

VR Server의 Kernel 구현을 위하여, interactive 3-D 환경 개발에 가장 많이 사용되고 있는 API인 VRCO의 CAVElib[18]를 사용하였으며 immersive interaction 환경에서의 input device tracking을 위해서는 trackd®[19]을 사용하였다. trackd는 "daemon" 프로그램으로 tracking device와 input device로부터 정보를 얻어서 immersive application 프로그램에서 위치정보를 사용할 수 있도록 한다.

Visualization 기능 개발을 위해서는 scientific visualization toolkit의 standard로 각광을 받고 있는 Object Oriented Visualization Toolkit인 VTK[2,15]를 사용하였다. VTK는 open source로 scalar, vector, tensor fields에 대한 다양한 visualization 방법들을 제공하고 module 간의 연결(VTK pipeline)을 구성함으로써 쉽게 visualization 결과를 얻을 수 있도록 한다. 그러나 두 library간의 design mismatch로 인하여 VTK를 CAVElib와 함께 사용하는 것은 쉽지 않다.



[그림 10] XMVR API Library

VTK를 CAVElib와 함께 사용하기 위하여 Rajlich(1998)는 "vtkActorToPF", Hall(1999)는 "vtk2CAVE", Shamonin(2002)은 “VtkCave"를 만들었다. "vtkActorToPF"는 VTK에서 polygonal data를 뽑아내어 이 데이터를 IRIS Performer(또는 OpenGL Performer)에서 랜더링하는 방법이고, "vtk2CAVE"는 display process가 shared memory에 복사된 vtkActor 정보에 접근하여 polygonal data를 OpenGL로 그리는 방법이다. 그리고 “VtkCave"는 VTK를 이용하여 CAVE application을 작성할 수 있도록 개발된 library로 Performer 사용에 독립적이다. Performer를 사용하지 않는 또 다른 방법으로 [17]에서는 application computation process가 VTK에서 계산된 결과를 polygonal data로 만들어서 이를 hard disk drive를 통해 display process들에게 전달하면 display process들은 OpenGL로 이 데이터를 그리는 방법을 사용하고 있는데 이 방법은 VTK의 polygon-based visualization 방법(예. Contour Lines, Stream Surfaces)에 잘 적용되는 것으로 알려져 있다. 본 연구에서는 CAVE application 개발이 용이하고 범용성이 뛰어난 VtkCave를 사용하였다([그림 10]).

[그림 11] XMVR VR Menu

1) Feature Handler에서 지원하는 기능

Feature Handler의 기능은 주로 [그림 11]의 메뉴에 의해서 제공되며 각 기능이 처리하는 내용은 다음과 같다.

가) Representation Change

데이터를 display하는 방법을 변환하는 기능으로 면(Surface), 선(Wireframe), 점(Points) 세 가지의 표현방법이 제공되고 있다.

나) Clipping

가시화된 object상에 절단면을 움직이면서 원하는 부분에서 절단면을 볼 수 있도록 하는 기능이다.

다) Slicing

개발중.

라) Probing

Point widget을 움직이면서 가시화된 object가 가지고 있는 scalar 값, 위치 정보 등을 interactive에게 보여준다.

마) Magnifying

3D로 display된 object를 다 방면에서 사용자가 세밀한 관찰을 할 수 있도록 obj

ect의 특정부분을 cube를 움직이면서 cube에 의해 절단된 부분을 자세히 볼 수 있도록 하는 기능이다.

바) Transformation(Translation, Scaling, Rotation)

가시화된 object에 대하여 위치이동, 확대/축소, 회전을 수행하는 기능이다.

2) View Manager에서 지원하는 기능

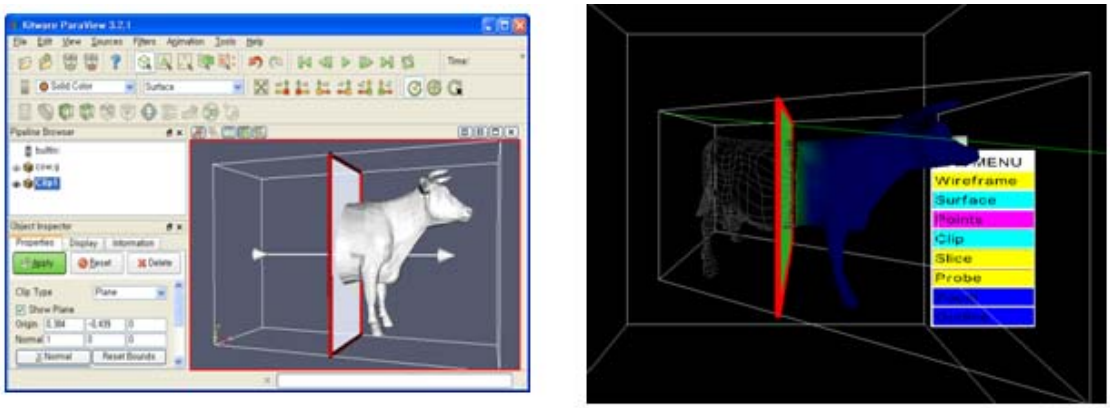
Multi-view를 지원하기 위하여 view manager는 사용자의 event가 발생한 view를 active 상태로 취급하고 나머지 view들은 inactive 상태로 설정한다. 모든 view는 각각 XMVR에서 지원하는 모든 기능을 동등하게 사용할 수 있으며 active view와 inactive view의 전환은, 단순히 사용자의 event를 마지막으로 받은 view를 기준으로 하고 menu에서 제공하는 기능은 항상 active view에 대해서만 적용된다.

4. Case Study: ParaView

VR Client의 visualization software로 ParaView를 적용한 결과는 다음과 같다.

- ParaView의 Clipping 기능 적용 결과

ParaView에서 "cow.g" 파일을 로딩한 후 Clipping Filter 적용하고 그 결과를 저장하였음. XMVR에서 저장된 가시화 결과를 로딩하여 Clipping 기능이 적용된 동일한 결과가 VR에서 바로 display 되었음.

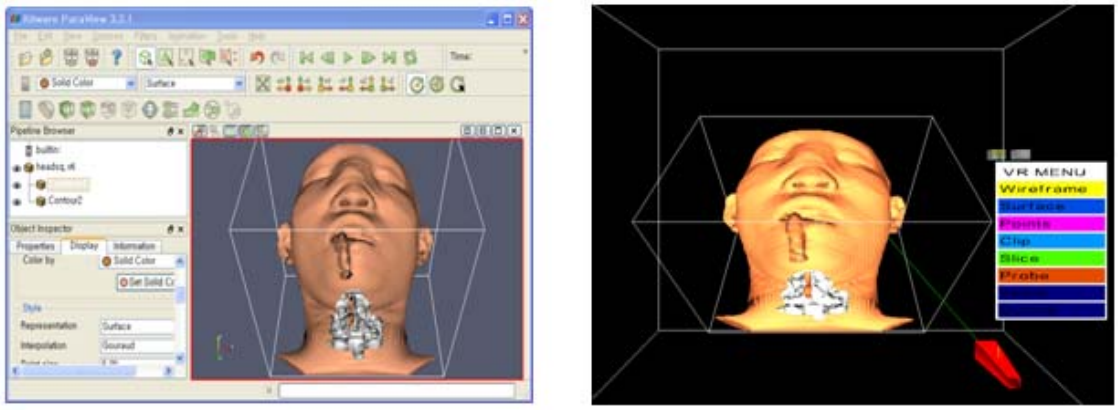


(a) (b)

[그림 12] (a) ParaView: Clipping filter, (b) XMVR 결과

● ParaView의 Contour 기능 적용 결과

ParaView에서 “headsq.vti" volume data를 loading한 후 skin과 bone surface를 추출하기 위하여 두 개의 contour filter를 적용하였음. 적용된 결과가 저장된 파일을 XMVR에서 loading하면 (b)와 같이 ParaView와 동일한 결과가 바로 VR에서 display됨.

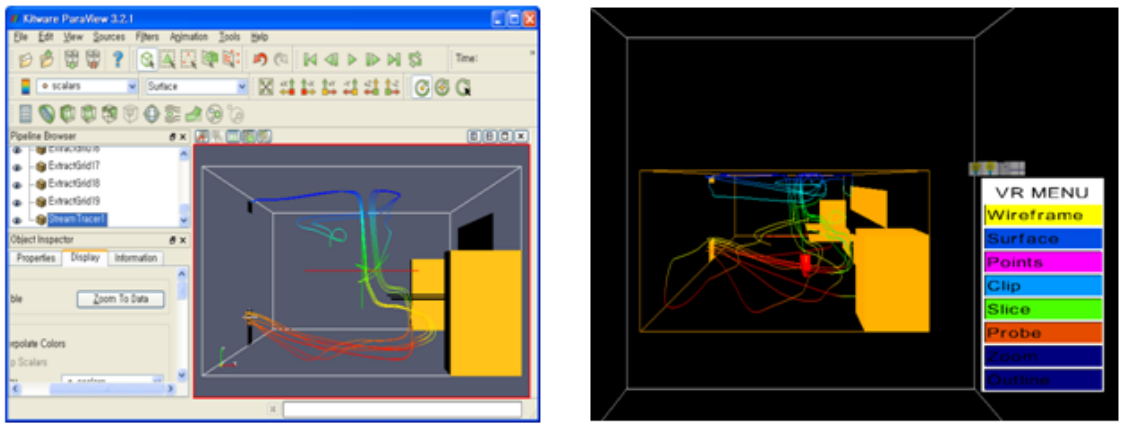


(a) (b)

[그림 13] (a) ParaView: Contour filter, (b) XMVR 결과

● ParaView의 StreamTracer 기능 적용 결과

ParaView에서 structured grid data format인 “office.vts"를 loading한 후 office desk, shelf 등을 display하기 위하여 19가지의 ExtractSubset filter를 적용하고 기체의 흐름을 파악하기 위하여 10개의 StreamTracer를 적용하였음. 가시화 결과가 저장된 파일을 XMVR에서 loading하면 (b)와 같이 ParaView와 동일한 결과가 바로 VR에서 display됨.

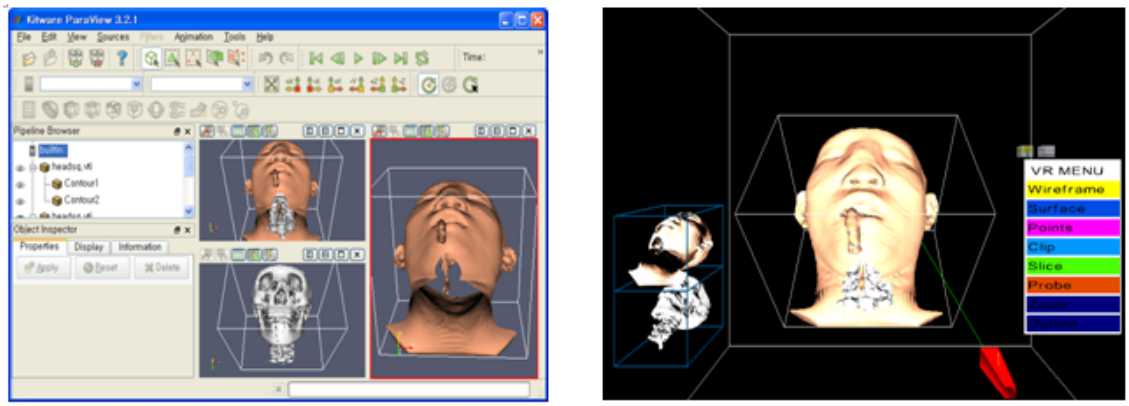


(a) (b)

[그림 14] (a) ParaView: ExtractSubset filter와 Stream Tracer filter 적용, (b) XMVR 결과

● ParaView의 Multiview 기능 적용 결과1

ParaView에서 동일한 volume data에 대해서 skin을 추출하는 view, bone을 추출하는 view, skin과 bone을 동시에 추출하는 view를 따로 생성한 후 각각 contour filter를 적용하고 그 결과를 하나의 파일에 저장하였음. 가시화 결과를 XMVR에서 loading하면 (b)와 같이 ParaView와 동일하게 여러 개의 view와 filter 적용결과를 동시에 지원한 결과가 VR에서 display됨.

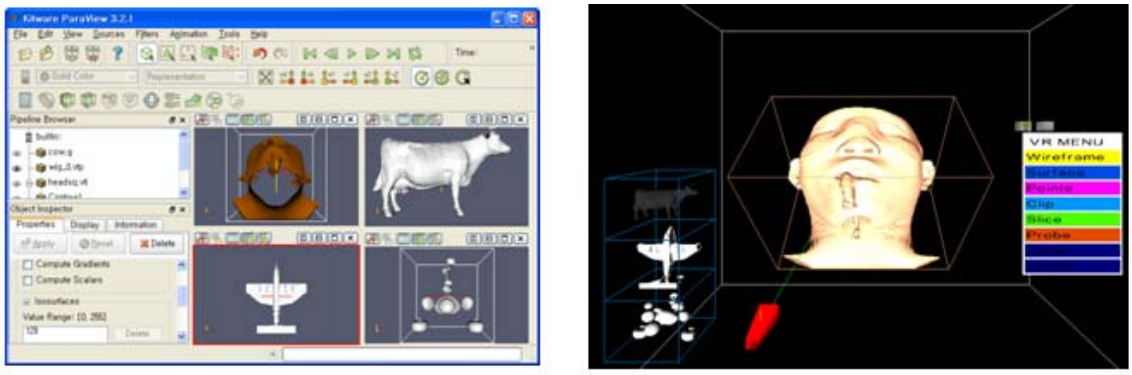


(a) (b)

[그림 15] (a) ParaView: ExtractSubset filter와 Stream Tracer filter 적용, (b) XMVR 결과

● ParaView의 Multiview 기능 적용 결과2

ParaView에서 서로 다른 네 개의 dataset에 대해서 각각 다른 view를 생성하고 가시화 실행한 후 결과를 저장하였음. 가시화 결과를 XMVR에서 loading하면 (b)와 같이 ParaView와 동일하게 여러 개의 view와 filter 적용결과를 동시에 지원한 결과가 VR에서 display됨.



(a) (b)

[그림 16] (a) ParaView: 네 개의 dataset과 multiview, (b) XMVR 결과

5. Conclusion

1. 더 많은 적용사례와 결과
2. 서로 다른 VR Client에 대한 Multi-view 지원
3. Feature Handler의 기능강화 : flow visualization 기능(사용자가 입력한 seed point 지원 포함), Volume visualization 등 더 많은 visualization 기능이 지원되어야 한다.
4. Collaboration 지원
5. 대용량 데이터의 빠른 VR display를 위한 병렬화

References

[1] Arjan J.F. Kok and Robert van Liere. A Multimodal Virtual Reality Interf

ace for 3D Interaction with VTK, Knowledge and Information Systems, Volume 13 , Issue 2 (October 2007), Pages: 197 - 219, Year of Publication: 2007

[2] W. J. Schroeder, K. M. Martin, and W. E. Lorensen. The design and implementation of

an object-oriented toolkit for 3D graphics and visualization. In IEEE Visualization '96,

pages 93--100, 1996.

[3] D. P. Shamoni. VtkCave.. http://staff.science.uva.nl/~dshamoni/manuals/VtkCave/index.html

[4]

[5] S. Bryson. Virtual reality in scientific visualization. Communications of the ACM, 39(5):62--

71, 1996.

[6] A. van Dam, A. S. Forsberg, D. H. Laidlaw, J. J. LaViola, and R. M. Simpson. Immersive vr

for scientific visualization: A progress report. IEEE Computer Graphics and Applications,

20(6):26--52, 2000.

[7] COVISE : http://www.hlrs.de/organization/vis/covise

[8] COVER : http://www.hlrs.de/organization/vis/covise/features/opencover/

[9] VisIT : https://wci.llnl.gov/codes/visit/home.html

[10] OpenDX : http://www.opendx.org/index2.php

[11] ParaView : http://paraview.org/New/index.html

[12] VisTrails : http://www.vistrails.com/

[13] VisTrails : David Koop, Carlos E. Scheidegger, Steven P. Callahan, Huy T. Vo, Juliana Freire and Claudio T. Silva. "Enabling Interactive Multiple-View Visualizations", To appear in IEEE Transactions on Visualization and Computer Graphics, 2008.

[14] SCIRun : http://software.sci.utah.edu/scirun.html

[15] Shroeder, W., Martin, K. and Lorensen, B. 2002 The Visualization Tool kit, 3rd Edition. Kitware Inc.,

New York.

[16] QUANTA : http://www.evl.uic.edu/cavern/quanta/

[17] N OHNO, A KAGEYAMA, K KUSANO "Virtual reality visualization by CAVE with VFIVE and VTK", Journal of Plasma Physics 72:0606, 1069-1072, Cambridge University Press, 12/2006.

[18] CAVElib : http://www.mechdyne.com/integratedSolutions/software/products/CAVELib/CAVELib.htm

[19] trackd : http://www.mechdyne.com/integratedSolutions/software/products/trackd/trackd.htm

[20] FlyBox : http://www.bgsystems.com/products/FlyBox.html

[21] IS-900 MicroTrax Devices : http://www.isense.com/uploadedFiles/Products/IS900%20MicroTrax%20datasheet.pdf

[22] AVS : http://www.avs.com/

[23] Gerwin de Haan, Michal Koutek, Frits H. Post, "Flexible Abstraction Layers for VR Application Development". IEEE Virtual Reality Conference 2007, March 10 - 14,

[24] D. Finkenzeller, M. Baas, S. Thring, S. Yigit, and A. Schmitt. Visum: A vr system for the interactive and dynamics simulation of mechatronic systems. In Proc. Virtual Concept 2003, Nov 2003.

[25] A. Backman. Colosseum: 3d-authoring framework for virtual environments.

In E. Kjems and R. Blach, editors, Proceedings of the 9th IPT and 11th Eurographics VE Workshop (EGVE) ''05, 2005.

[26] M. Goslin and M. R. Mine. The Panda3D graphics engine. Computer, 37(10):112--114, 2004.

[27] J. Springer, H. Tramberend, and B. Fr ¨ohlich. On scripting in distribut

ed
virtual environments. In Proceedings of the 4th IPT Work-
shop, June 2000.