# 가상환경의 저작을 위한 시각 프로그래밍 기법

박성준 °, 김지인
건국대학교 컴퓨터공학과

# A Visual Programming Method for Authoring Virtual Environment

Sung-Jun Prak °, Jee-In Kim
Department of Computer Science & Engineering Kon-Kuk University

## 요 약

가상환경기술이 여러 분야에서 보편화되고 활용되기 시작하면서 컴퓨터 프로그래밍 기술이 없어도 가상환경을 손쉽게 꾸밀 수 있는 저작기술이 필요하게 되었다. 본 논문에서는 가상환경 저작용 시각 언어인 Virtual Environment Specification Language(VESL)를 사용하여 컴퓨터 프로그래밍 코딩을 하지 않아도 가상환경을 저작할 수 있는 기술을 구현하였다. 우리가 개발한 시각언어 편집기는 VESL을 기반으로 하여 3차원 가상환경을 구축하고 가상환경 내에 정의된 객체들의 물리적 속성을 정의하고 객체들간의 관계 및 유지를 명시함으로써 보다 현실감 있는 동적인 가상 환경을 손쉽게 저작할 수 있도록 하였다.

## 1. 개 요

컴퓨터 그래픽스 하드웨어와 소프트웨어 기술이 발전하면서, 가상현실 (Virtual Reality) 세계를 구축하고, 가상의 공간 속에서 연구, 개발, 교육, 훈련, 치료, 등의 활동을 수행하는 기술이 실용화되어 가고 있다. 특히 Sense8사의 WorldToolKit이나 Division사의 dVISE같은 가상현실 저작도구의 출현으로 인하여, 3차원 그래픽스나 가상현실 기술에 대한 전문지식이 없는 일반 사용자도 가상현실 세계를 손쉽게 만들 수 있게 되었다. 가상환경을 구축하는데 있어서 필요한 핵심 기술 중의 하나는 가상환경을 저작하는 기술이다.

가상환경 저작 방법은 배우기 쉽고, 쓰기가 편해야 한다. 즉, 컴퓨터 프로그래밍이나 컴퓨터 그래픽스에 관한 전문 지식이 거의 없는 사람들도 가상현실 환경을 설계, 제작 및 분석할 수 있도록 해야 한다. 본 논문에서 구현한 시각언어 편집기는 시각적 설계와 논리적 설계를 함께 실행할 수 있는 가상환경 저작도구 툴(VESL Editor)이다. VESL Editor는 프로그래밍 작업을 시각적으로 수행할 수 있는, 사용자 중심의 시각 프로그래밍 환경을 제공함으로써 VR 설계자로 하여금 가상 환경을 보다 쉽게 구축하도록 하였다. VESL Editor는 정적인 가상환경을 설계할 수 있을 뿐만 아니라 동적인 가상세계를 다룰 수 있도록 하였다. 즉, 충돌검사(Collision Detection), 객체간의 Constraints의 표현 및 물체의 움직임을 정의하는 다양한 Tasks를 처리할 수 있는 기능을 갖추게 된다. 또한, VESL Editor를 사용하여 가상의 3차원 건축 공간을 꾸미고, 그 내부를 항해할 수 있는 Architecture Walkthrough를 위한 가상환경을 구축하려 한다.

본 논문의 구성은 다음과 같다. 2장에서는 가상환경 저작 툴에 대한 관련 연구를 소개하고, 3장에서는 본 논문에서 제안한 VESL Editor에 대한 전반적인 개념과 기능들을 다루고 있다. 4장에서는 VESL Editor의 구현에 대해서 소개하고, 마지막으로 결론 및 향후 과제에 대해서 논의하였다.

## 2. 관련 연구

기존의 대표적인 가상환경 저작 도구들 중에는 Division의 dVISE, Sense8의 WorldToolkit, 그리고 Superscape의 VRT 등이 있다. SuperScape Virtual Reality Toolkit(VRT)는 Dimension Internation사에서 상업용으로 판매하는 제품으로, Shape editor, World editor Console editor, Visualizer로 구성된다. 각 Editor는 가상환경을 쉽게 저작할 수 있도록 System에서 제공하고 있다. 각 물체의 모양을 정의할 수 있으며, 그 물체들을 가상세계 내에 배치하고, 여러 가지 메타포어를 통해 다양한 시각을 정의할 수 있다. WorldToolKit(WTK)는 Sense 8사에서 개발한 것으로, VR 응용프로그램을 작성하기 위한 C 함수 라이브러리의 모임이다. Rend386에서는 가상 세계 제작을 위한 시스템을 따로 제공하지 않고 text editing을 통하여 직접 각 좌표를 비롯한 형상 정보 등을 입력하여 데이터 베이스를 생성하도록 하고 있다. 따라서 복잡한 가상 세계를 제작하는데는 많은 시간과 노력이 필요하다. Carnegie Mellon University에서 개발한 Alice는 스크립트와 프로토타입 환경의 어플리케이션으로써 3D 객체에 대한 동적인 속성을 처리할 수 있도록 만들어진 가상환경 저작 도구이다. Alice의 특징은 배우기 쉽다는데 있다. 추가된 객체들에 대해서 적절하게 위치시키기만 하면 하나의 가상환경을 쉽게 구축할 수 있다.

이러한 툴킷(Toolkit)들은 대부분 시각적 설계에 중점을 두고 있으

며 본 논문에서 구현한 VESL Editor는 이들과 비교해 볼 때 몇가지 다른 점이 있다. 첫째, 기존의 툴들은 객체들간의 시각적인 관계나 논리적인 관계가 시각적으로 나타나지 않는다는 점이다. 둘째, 물체 들간의 제약조건(Constraint)을 일관성 있게 유지하는 시스템이 제공되지 못한다는 점이다. 마지막으로, 동적인 가상 환경을 만들기 위해 들리적인 Task를 직접 프로그램 언어를 사용하여 코딩해야 한다는 이려움이 있다. 본 논문에서는 이러한 문제점들을 사용자 측면에서 해결할 수 있는 기법을 소개한다

## 3. VESL Editor

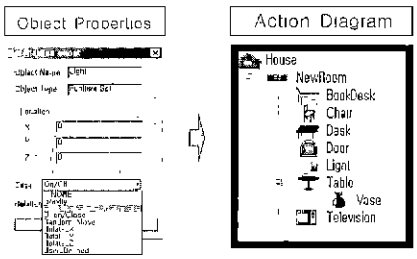
### 3.1 개요

VESL를 이용하여 가상환경을 꾸미려면, 시각언어 VESL를 위한 대화형 Editor가 필요하다. 가상환경에 필요한 그래픽 물체들은 미리 제작되어서 아이콘 형태로 저장되어 있는데, VESL Editor를 사용하여 설계하는 사람은 필요한 아이콘을 갤러리에서 골라서 가상공간을 꾸민다. 또한, VESL Editor에는 기존의 Modeling Editor의 기능에 더하여 물체에 대한 동적인 속성과 논리적 속성 및 물체간의 관계인 Constraint를 시각적으로 정의할 수 있어야 한다. 만들어진 가상공간을 3차원으로 살펴 보고, 애니메이션과 시뮬레이션을 통하여 가상공간의 동작을 점검해 볼 수 있어야 한다

VESL Editor의 특징으로는 가상공간의 시각적 설계를 위한 Space Diagram과 논리적 설계를 위한 Action Diagram이라 2개의 Panel을 지공하고 있다는 점이다. VR 설계자는 VESL Editor를 이용하여 가상환경을 설계하기 위하여 논리적 설계는 Action Diagram에 그리고 시각적 설계는 Space Diagram에 표현한다. Action Diagram과 Space Diagram은 서로 일관성을 유지하여야 한다. 이 조건을 만족시키기 위해서 VESL Editor는 이 두 Diagram의 내용이 서로 일치하도록 항상 점검하고 보완해 주어야 한다. Space Diagram에서 시각적 설계를 마치게 되면 VR 설계자는 Action Diagram으로 이동하여 필요한 논리적 설계를 실시한다. VESL Editor를 이용하여 가상환경을 모델링 한 후에는 Viewer를 이용하여 3차원 가상환경을 시뮬레이션함으로써 가상환경의 타당성을 점검해 볼 수 있다.

### 3.2 Action Diagram

Action Diagram은 VESL을 이용하여 저작된 가상환경의 논리적 설계를 도리형식으로 시각화하여 표현한 것이다. 논리적 설계를 이루고 있는 아이콘들은 계층적으로 구성되어 있다. [그림 1]은 Action Diagram을 그리기 위하여 Object Property에서 Task를 선택하는 과정을 나타낸 그림이다. VESL Editor에서 제공하고 있는 Task는 문을 열고 닫는 Open, Close를 비롯하여 Sound, Translation, Scale Rotate Gravity등 다양하게 정의되어 있다




< 그림 1. Task Properties과 Action Diagram >

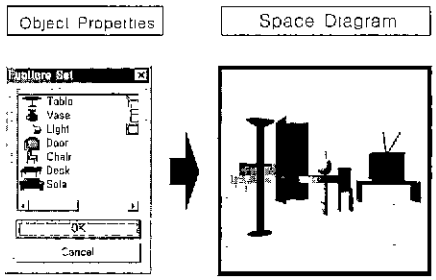
VESL Editor는 물체들 사이의 관계를 Constraint로 표현하고 객체들 사이의 관계나 혹은 객체와 작업 사이의 관계를 정의한다. 본 연구는 Constraint 기반의 모델링 파라다임(Constraint-based Modeling Paradigm)에 근거하고 있으며, 정의된 Constraint를 유지하기 위하여 Constraints Maintenance System을 제공하고 있다 [2] Constraint를 사용하여 Visual Programming기법을 연구한 예로는 Borning의 ThingLab[9]과 Nelson의 Juno를 들 수 있다. 이 두개의 시스템은 2차원적인 graphic modeling에 기반을 두고 있다. 또한 3차원 객체에 대한 Constraint를 다루고 있는 Sistare의 Converge system도 좋은 보기이다. VESL Editor는 2차원적인 특징과 3차원적인 특징을 모두 포함하고 있기 때문에 위의 도구들이 갖는 특징을 모두 갖추게 하였다. Constraints Maintenance System은 정의된 객체에 대한 제약조건을 모두 만족시키고 유지할 수 있도록 하고 있다. Constraint Management System은 다양하게 변화하는 데이터의 정보를 일관성 있게 처리하여 주어진 제약조건을 항상 만족 시킬 수 있도록 유지해야 한다. 그 결과 VESL Editor는 사용자들이 사용하기에 보다 편리하고, 배우기 쉬운 GUI를 제공하고 있다

### 3.3 Space Diagram

VR 설계자는 Space Diagram을 그려서 시각적 모델을 설계한다. Space Diagram은 3차원 view로 구성된 GUI 환경을 제공하고 있으며, 프로그램 비전문가도 Object Icon을 사용하여 가상공간을 구축할 수 있다. 본 연구에서 응용하고자 하는 건축 시뮬레이션 분야인 Architecture Walkthrough를 위해 일반적으로 채택되는 오브젝트를 3부류로 분류하고 나타내면 다음과 같다

- FurniSet : Bed, Desk, Chair, Computer,
- kitchenSet : Sink_Table, Table, Chair,
- BathSet : Shower_Bath, Chamber_Pot,

위에서 제시된 아이콘들은 VESL 관리자가 만들어서 물체들에 대한 특성을 대표할 수 있도록 시각화하여 놓았으므로 VR 설계자는 갤러리에서 아이콘을 선택하여 물체를 그린다. 선택된 객체는 Space Diagram에 생성되며, 동시에 Action Diagram의 트리에 첨가된다. 사용자는 Space Diagram에서 생성된 객체에 대해 적당한 위치를 잡아가면서 모델 하우스를 모델링 한다. 완성된 객체들에 대한 속성들은 자동적으로 VESL Editor System에 저장된다. 이러한 Object Icon이 나타내는 시각적 속성에는 Light Source, Size, Position Viewpoint 등이 있다. VR 설계자는 이 같은 객체들에 대한 속성들을 변화시키 주면서 보다 현실감 있는 가상공간을 구축할 수 있다



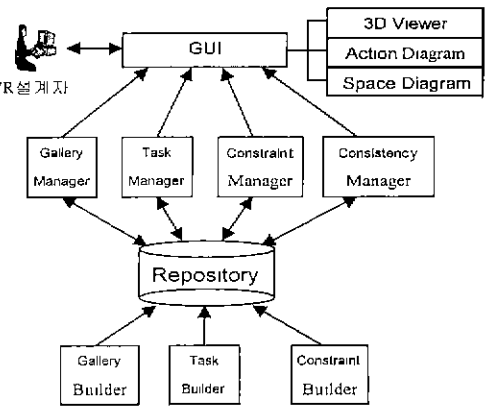

< 그림 2. Object Icon과 Space Diagram >

[그림2]는 Object Icon과 Space Diagram을 보여주고 있다. View

는 3D View를 제공함으로써 VESL 설계자로 하여금 보다 현실감 있는 가상공간을 제작할 수 있도록 환경을 제공한다. View 내에서 객체를 마우스로 클릭하면 객체에 대한 속성을 보여주는 다이얼로그 박스가 나타나게 된다. 객체에 대한 속성에는 이름과 타입, 좌표값, Task, Relationship등의 정보가 정의되어져 있고, VR 설계자는 이러한 값들을 재 정의하면서 만족스러운 가상공간을 만들어 낼 수 있다.

## 4. 구현

VESL Editor는 Visual C++5.0과 Window MFC 라이브러리를 이용하여 구현하였으며, 3차원 뷰어(Viewer)는 OpenGL로 구현하였다. 또한 이 시스템은 Windows 95와 Windows NT환경에서 실행되며, VR 설계자가 가상공간을 보다 쉽게 구축 하는 데에 필요한 Object Editor, Task Editor, Icon Editor를 가지는 Gallery Builder를 개발하였다. [그림3]은 VESL Editor System의 Architecture를 나타낸 그림이다. 우선 사용자는 VESL Editor의 GUI를 통하여 가상환경을 구축한다. 논리적 설계에 해당하는 Action Diagram과 Visual 설계에 해당하는 Space Diagram을 이용하여 가상 건축 공간을 설계한다. 이렇게 설계된 가상세계는 3D 뷰어를 거쳐 확인하는 단계의 과정이 이루어진다.




< 그림 3 VESL Editor System Architecture >

GUI는 VESL Editor에서 제공하는 4개의 Manager에 의해서 VR 설계자에게 보여주게 된다. Gallery Manager, Task Manage Constraint Manager, Consistency Manager 들은 가상환경 데이터베이스(Repository)로부터 필요로 하는 정보를 가지고 와서 VR 설계자에게 전달해 주는 역할을 담당한다. Gallery Builder는 3D 응용 어플리케이션을 통하여 생성된 객체를 OpenGL 코드로 변환시켜 주는 Application에 해당하는 System이다. Task Builoer는 VR 설계자가 구축한 가상공간에다 동적인 속성을 부여할 수 있도록 모듈화된 라이브러리를 생성해주는 시스템이다. Task 에는 Move, Rotation Gravitation, 등이 있으며, 객체에 대한 속성을 정의하고 있다. 모듈화 되어있는 이러한 Task들은 Action Diagram에서 생성된 객체들에 대해 독립적으로 적용할 수 있으며, Interative하게 3D 뷰어에서 확인 할 수 있다. 이러한 Task들은 OOP(Object Oriented Programming)의 개념[5][6]을 도입하여 Polymorphism과 Inheritance를 사용하고 있다. 예를 들어 Open(Open-U, Open-R) 문에 대한 Task인데 여닫이문과 미닫이문에 다같이 적용될 수 있다. Open은 적용되는 문의 성격에 따라서 행동양식이 달라진다. 즉, 회전하여 열리거나, 오른쪽으로 이끌려지면서 열리게 된다. 또한, 상위계층의 속성은 하위계층의 속성(Size, Color)은 하위 계층인 객체에도 적용된다. 이러한 상속성은 건축 설계의 공통된 특징을 Class화 함으로써 적용될 수 있다. Constraint Builder는 가상공간 내에서의 객체들 사이의 관계(Relationship)를 명시할 수 있도록 도와준다. 객체간의 관계는 IN, ON, OVER등이 있으며, Constraint Manager에 의해서 유지된다. VR 설계자는 객체간의 관계를 정의할 수 있으며, 한번 정의된 관계는 Constraint Maintenance에 의하여 계속 유지된다. Consistency Manager는 Action Diagram과 Space Diagram사이의 내용을 일관성 있게 검사하고 유지하는 시스템이다.

## 5. 결론 및 향후과제

본 논문에서는 기존의 VR 저작 도구들이 대부분 시각적 설계에만 중점을 두었다는 점과 가상환경을 저작하기 위하여 프로그래밍 기술을 필요로 한다는 점을 해결하기 위해 새롭게 만든 시각언어(VESL Visual Environment Specification Language)를 가지고 가상공간을 만들 수 있는 Editor를 개발하였다. VESL Editor는 객체들간의 논리적 특성과 관계를 시각화하고 그들간의 관계를 계속적으로 유지 할 수 있도록 Constraint Maintenance에 기반을 두고 있다. VESL Editor는 다양한 물리적 속성을 정의하고 있는 Task를 제공하고 있으며, 이러한 행동양식을 이용하여 보다 동적인 가상환경을 만들어 낼 수 있다. 앞으로 향후 과제로는 이러한 행동양식을 스크립트화하여 사각화 할 예정이다. 스크립트화 한 행동양식들은 하나의 시나리오를 만들어 낼 수 있으며, 여러 개의 시나리오를 조합하여 커다랗고 동적인 가상공간을 만들어 낼 수 있다. 마지막으로 VESL Editor에서 제작한 가상공간을 VRML Code로 전환할 수 있는 번역기를 만들 예정이며, VRML 번역기를 통해 인터넷 가상환경의 저작을 쉽게 도와주는 도구를 개발할 예정이다.

## 참고문헌

[1] Schrr, A Winter, A Z ndorf, "Visual Programming with Grap Rewriting Systems", IEEE, pp 1049-2615, 1995

[2] Rich McDaniel and Brad A.Myers Amulets Dynamic an Flexible Prototype-Instance Object and Constraint System C++, UIST96, pp 176, July, 1995

[3] Alan Borning, Robert Duisberg and Michael Maher Constrai hierarchies, in OOPSLA87 Conference Proceedings, ACM pp.48-60, October, 1987

[4] Amarnath Gupta, Ramesh Jain, "Visual Information Retrieva Communications of the Acm, vol 40, No 5, May, 1997

[5] Jock Mackinlay, "Automating the Design of Graphic Presentations of Relational Information", ACM Transactio on Graphics, Vol 5, No 2, pp 110-141, April, 1986

[6] Roger B Dannenberg, "A Structure for Efficient Updat Incremental Redisplay and Undo in Graphical Editors Software-Practice and Experience, Vol (2), pp 109-13 Feb 1992

[7] Kent Wittenburg, Louis Weitzman and Jim Talley "Unification-based Grammar and Tabular Parsing f Graphical Languages", Journal of Visual Languages an Computing 2, pp 347-370, 1991

[8] Eric J Golin and Steven P Reiss, "The Specification o Visual Language Syntax", Journal of Visual Languages an Computing 1, pp 141-157, 1990

[9] Eric J.Golin, "Parsing Visual Languages with Picture Layo Grammars", Journal of Visual Languages and Computing pp. 371-393, 19