3차원 게임을 위한 시맨틱 가상환경 생성과 네비게이션 제어

장 현 덕[†] · 이 재 문[††] · 이 명 원[†††]

요 약

3차원 게임 프로그래밍에서 요구되는 가상환경 생성에 있어서 가상환경의 기능이 단순한 배경 제공의 기능 외에 실세계 환경과 직접적으로 관련되는 정보를 포함하는 것이 필요하다. 본 연구에서는 게임에서 실제 환경을 가시화한 가상환경에서 캐릭터가 진행할 때 가상환경에서의 지리적 위치를 알게 함과 동시에 환경과 관련된 정보를 실시간으로 제공하는 시스템을 구현한다. 이러한 기능은 가상환경이 특정 지역에서 고유한 정보를 보유할 수 있게 하고 환경의 조건에 따라 장면을 제어해갈 수 있는 점에서 시맨틱 가상 환경(Semantic Virtual Environment) 구현의 한 방법을 제공한다고 할 수 있다. 본 논문에서는 이러한 게임에서의 시맨틱 가상환경 구현을 목적으로 유비쿼터스 환경에서 위치기반 실시간 정보 입력을 가능하게 하는 시스템과 가상환경 내 특정 위치를 찾아 대화형으로 네비게이션을 제어할 수 있도록 해주는 방법에 대해 설명한다.

키워드: 가상환경, 게임 가상환경, 게임 네비게이션 제어, 시맨틱 가상환경, 게임 엔진

Semantic Virtual Environment Generation and Navigation Control for 3D Games

Hyun Duk Jang[†] · Jae Moon Lee[††] · Myeong Won Lee[†††]

ABSTRACT

In conventional game systems, virtual environments usually have just the role of a background without the direct relationships for game characters. They do not consider the semantics about virtual environments. In this paper, we develop a game navigation system that provides semantic information about virtual environments including geographical, historical or any other location-dependent information. Then, the game character obtains the geographical location and its related information when it navigates through a virtual environment. It can be an implementation method for a semantic virtual environment because it can have the environment maintain its semantics depending on the specific location. In addition, we describe a method that can control a character's motion in the semantic virtual environment interactively, and that can input specific information according to the location of the character.

Key Words : Game Virtual Environment, Game Navigation Control, Semantic Environment, Game Engine, 3D Game Control

1. 서 론

3차원 게임 프로그래밍 기술은 3D 기반의 각종 시뮬레이션 표현 기법을 포함하고 있으며 가상환경의 가시화 프로그램 개발이 기본적으로 필요하다. 이러한 게임 프로그램에서 사용자가 실제 게임과 접하게 되는 인터페이스인 클라이언트 어플리케이션 프로그램은 게임 로직 모듈과 게임 엔진 모듈로 구성된다. 게임 로직 모듈은 게임 시나리오에 따라 게임을 진행하는 로직이 들어 있는 모듈로, 게임 내에 존재하는 캐릭터, 주변 사물, 지형, 건물 등의 객체가 어떻게 그려질지에 관해서는 관여하지 않고 어떤 객체가 그려질 것인가에 대해서만 관여한다. 다시 말해서 게임의 전체 흐름을 통제하는 모듈이다. 따라서 이 모듈은 하드웨어, OS 혹은 API에 종속되지 않는 모듈에 해당한다. 이렇게 종속되지 않게 하기 위해서는 OS나 API에 관한 인터페이스를 제공해 주어야 하는데, 이것이 게임 엔진 모듈이다[1][8].

게임 엔진 모듈은 게임 로직에서 나타나는 모든 객체들의 행위 및 표현에 필요한 모든 작업을 담당하게 된다. 보통 게임의 배경으로 나타나는 3차원 가상환경 표현 기능도 게임 엔진 모듈에 포함된다. 이러한 게임 엔진 모듈에는 렌더링 라이브러리, 지형관리, 객체 관리, 리소스 관리, 애플리케이션 프레임워크, 네트워크 모듈 등 여러 모듈들로 구성되어 서로 유기적으로 작동하면서 게임에 필요한 기능을 제공

† 준 회 원 : 인하대학교 정보공학과 대학원생
†† 정 회 원 : 수원대학교 컴퓨터학과 졸업
††† 정 회 원 : 수원대학교 컴퓨터학과 및 인터넷정보공학과 부교수(교신저자)
논문접수 : 2005년 9월 20일, 심사완료 : 2007년 6월 18일

한다[2].

이제까지 주로 게임의 배경으로서 표현되어 왔던 3차원 가상환경은 게임 캐릭터 이동에 의해 발생할 수 있는 충돌 처리 구현 외에는 캐릭터와 관련되어서 특별한 기능을 부여하고 있지는 않고 있다[7][9]. 본 연구에서는 3차원 가상환경이 이러한 배경 제공의 기능 외에도 지리적 실제 위치와 관련하여 게임 로직 구현에 영향을 주는데 사용할 수 있도록 지리적 의미가 있는 정보를 제공할 수 있는 환경 정보 입력 인터페이스를 개발하였다.

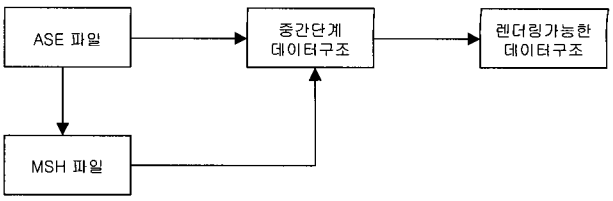
본 연구에서는 이상과 같이 실제 가상환경에 지리적 혹은 역사적 의미를 부여하여 게임의 캐릭터가 네비게이션 중에 환경에 의한 영향을 받을 수 있게 하거나 환경과 관련한 정보를 얻을 수 있게 하였으며, 이와 같이 게임 로직 구현에 환경 정보를 사용할 수 있게 해주는 기능을 가지는 가상환경을 시맨틱 가상환경이라고 부른다. 이것은 최근 데이터베이스 분야에서의 시맨틱 웹이나 시맨틱 DB와는 다른 개념이라고 할 수 있다[3][12].

본 연구에서는 게임 어플리케이션 개발에 있어서 공통적으로 필요한 기능인 3차원 시맨틱 가상 환경의 생성에 필요한 위치기반 정보 입력기와 이러한 가상환경에서 게임 캐릭터의 네비게이션을 제어하는 기능 개발을 중점으로 설명한다. 여기서 위치기반 정보 입력기란 실제 환경을 게임 가상환경으로 구성하였을 경우에 어떤 특정 위치와 관련된 정보를 게임 중에 실시간 입력 가능하게 하는 도구를 의미한다. 이를 이용하여 가상환경에 의미를 부여할 수 있도록 하는 시맨틱 가상환경을 구현하였다.

2. 게임 가상환경 생성

본 절에서는 게임의 배경이 되는 3차원 가상환경 생성 방법에 대해서 기술한다. 게임 가상환경을 구성하는 구성요소에는 건물, 지물 그리고 지형이 존재한다. 보통 건물과 지물은 일반 그래픽스 모델러에서 제작된 뒤에 게임 프로그래밍 환경으로 읽어들인다. 일반 모델러에서 제공하는 메쉬 정보를 이용하여 게임 프로그래밍에서 가상환경 표현을 위한 기하나 속성 정보로 사용하게 된다. 예를 들어, 3DS MAX에서는 ASE(Ascii Scene Export)를 제공하기 때문에 메쉬 정보를 텍스트 파일로 얻을 수 있다. 이와 같은 텍스트로 표현되는 메쉬 정보는 다루기 쉽고, 수정이 용이하나, 이 파일에는 응용에 따라 직접 사용하지 않는 정보가 많이 포함되어 있고, 파일 크기가 매우 크다는 단점이 있다. 예를 들어, 본 연구에서 생성한 가상환경의 일부인 경복궁을 모델링한 메쉬 정보 파일은 약 12메가 바이트정도 차지하였는데, 이렇게 큰 크기의 메쉬 정보 파일은 게임을 실행할 때 문제가 된다. 게임을 실행할 때마다 이 파일을 로드하여 파싱하는 과정을 거쳐 그래픽스 라이브러리에서 사용할 수 있는 데이터 타입으로 재구성하는 데에는 많은 시간이 걸리기 때문이다.

그래서, 본 연구에서는 MSH 라는 고유의 바이너리 파일 포맷을 고안하여 텍스트 파일보다 작은 크기를 가지며, 메쉬 변환에서도 상당한 성능 향상을 얻게 되었다(그림 1). 메쉬 정보를 로드하는 과정 중에서 파싱하는 작업이 가장 오래 걸리기 때문에 이 과정을 생략하게 된다. 본 연구의 메쉬 관리 모듈에서는 ASE 파일을 MSH 파일로 변환해주는 별도의 애플리케이션을 제공하여 클라이언트 애플리케이션에서 사용할 수 있도록 하였다.



(그림 1) 가상환경 메쉬 데이터 생성

가상환경 모델링 중에서 지형은 앞에서의 다른 모델과는 달리 일반 모델러로 저작되지 않고 다른 방법을 사용하여 렌더링한다[4]. 본 연구에서는 ROAM(Realtime Optimally Adapting Meshes) 방식을 사용하여 랜더링 하였다[5]. ROAM은 지형을 표현하는데 있어서 LOD (Level Of Detail)을 구현하는 알고리즘 중 하나이다. ROAM에서는 동적으로 폴리곤 수를 조절하는 기능을 제공해서 매 프레임마다 렌더링해야 할 폴리곤을 테셀레이션하게 된다. ROAM은 일정 영역으로 구분되는 패치단위로 테셀레이션을 하는데, 각각의 패치에서 테셀레이션 레벨이 다르면 메쉬가 깨져버리는 문제가 발생한다. ROAM에서는 모든 삼각형에 대한 이웃한 삼각형을 가지고 있어서 테셀레이션할 때 이웃한 삼각형도 테셀레이션함으로서 이러한 문제를 해결한다.

배경 및 지형 등의 가상환경이 제작된 후에는 연속적으로 이 가상환경과 캐릭터를 이용하여 게임 컨텐츠를 제공해야 하는데, 일반 그래픽스 도구로 제작한다는 것은 매우 힘든 일이다. 캐릭터가 가상환경 상에서 게임 시나리오에 따라 움직이도록 해주고 이 때 발생할 충돌 처리를 사전에 조정해주는 역할을 하는 맵 에디터가 필요하게 된다. 이 도구는 사용자가 가상환경을 게임 내용에 맞추어 쉽게 저작하는데 목표를 두고 있으며, 이것을 이용하여 다음 두 종류의 가상공간을 저작할 수 있다.

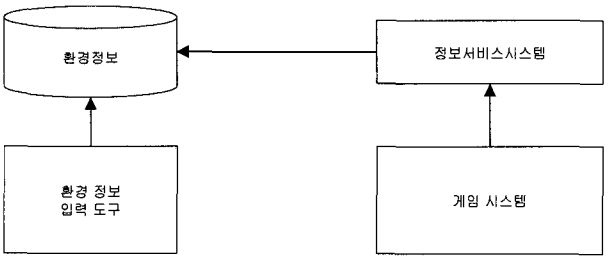
하나는 실내 공간으로 이 실내 공간을 저작할 때는 일반 그래픽스 도구로 저작된 내부 공간을 읽어 들여서 게임 엔진에 최적화된 데이터 구조로 실내 공간을 저장하는 기능을 갖는다. 또한 실내 공간 내에 정적으로 위치하는 객체를 배치하는 기능을 갖는다. 또한 실내 공간의 BSP (Binary Space Partition)[4] 와 PVS (Potentially Visible Set)을 생성하여 엔진이 더 효율적으로 실내 공간을 렌더링 할 수 있게 한다[5]. 또 다른 기능은 실외 공간을 저작하는 기능인데, 지형의 높낮이와 지형 타일의 텍스춰를 설정하고, 랜덤 지형 생성 및 raw 비트맵 파일을 통해 초기지형을 생성할 수 있다. 실내 공간 저작과 마찬가지로 실외 공간에 존재하는 정적인 객체를 배치시킬 수 있다. 또한 방대한 크기의 실외 공간을 표현하고, 관리하기 위해서 ROAM 기법을 사용한다.

3. 시맨틱 가상환경 정보 구성

기존의 게임 가상환경은 등장하는 캐릭터와의 관계에 있어서 가상환경을 캐릭터와는 독립적으로 모델링과 렌더링 표현에만 중점을 두어 개발되었다. 이에 비해 본 연구에서는 시맨틱 가상환경 구현의 일환으로 실제 지리적 의미를 갖고 객체 상호간의 관계나 위치 기반의 환경 정보를 정의 및 보유할 수 있도록 하였다.

가상환경에서의 지리적 위치와 관련 정보를 캐릭터의 움직임과 동기화시키기 위해서는 먼저 지리적 위치에 대해 가상환경 정보가 생성, 저장 및 관리되어야 한다. 이것은 가상환경에서의 표현이 실세계 지형의 사실적 표현도 중요하지만 지형의 물리적 혹은 지리적 특성과 역사적 정보 등의 의미있는 정보를 제공하기 위해서 필요하다. (그림 2)는 게임 프로그램과 정보서비스 시스템간의 관계를 보여준다.

본 연구에서는 이와 관련하여 게임 내 가상환경에서 캐릭터가 지나가는 임의의 위치에 정보를 입력할 수 있는 실시간 환경정보입력기를 개발하였다(그림 3). 이것은 게임 캐릭터가 네비게이션 중에 (x, z) 좌표 지점에서 지형과 관련한 정보를 화면에 디스플레이되도록 해주기 위하여 해당 위치에 텍스트 정보를 입력하는데 사용하도록 개발된 인터페이스이다. 이 그림에서 바탕의 2차원 그림이 게임 가상환경의 XZ 평면에 해당하며 다이얼로그 박스에서 해당 위치의 정보를 입력하게 된다. 이 도구를 이용하여 게임 가상환경을 실세계 환경과 동일하게 구성하고 실세계 해당 위치의 지리적 혹은 역사적 정보를 입력하여 가상환경 내에서 캐릭터의 이동 위치에 따라 실시간으로 지리상의 정보를 얻을 수 있도록 하였다.



(그림 2) 환경 정보와 게임 시스템

(그림 3) 환경 정보 입력기

4. 가상환경 네비게이션 제어

캐릭터가 가상환경 내에서 사용자의 의도에 따라 움직임을 진행하는 네비게이션 실행을 위해서는 캐릭터와 카메라에 의한 이동이 필요하다. 캐릭터는 현재의 위치, 방향, 속도를 가지며 이러한 값들로 키보드 입력에 따라 캐릭터를 이동시킨다. 회전이 필요한 곳에서는 마우스의 움직임에 따라 Yaw 각을 얻어낸 후 월드 좌표계에서의 Y 축을 기준으로 Yaw 각도만큼 캐릭터의 방향을 회전시킨다[4]. 후에 회전된 캐릭터의 방향과 월드 좌표계의 Y축과의 외적을 계산한다. 이 결과는 캐릭터의 오른쪽 방향을 나타내는 벡터가 되며 이 벡터를 쿼터니온을 이용하여 Pitch 각도만큼 회전시키게 된다.

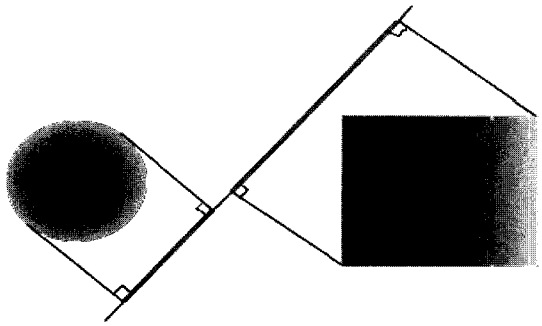
카메라의 이동은 현재 캐릭터의 위치에서 위의 과정으로 계산된 캐릭터의 정규화된 방향 벡터를 빼준 위치에 카메리를 위치시키고, 초점을 캐릭터의 방향으로 맞춘다. 이렇게 하면 카메라가 항상 캐릭터 뒤를 따라다니게 할 수 있다.

캐릭터의 네비게이션을 위한 키 종류와 각 기능에 있어서 기본적으로 키보드와 마우스를 사용한다. 키보드는 캐릭터의 이동을 조종하며 마우스로 캐릭터의 방향을 바꾼다. 키보드로는 캐릭터를 앞/뒤 혹은 좌/우로 이동시킬 수 있고, 이 방향의 기준은 현재 캐릭터의 방향이 된다. 이에 해당하는 키로 각각 앞[W], 뒤[S], 좌[A], 우[D]가 사용되었다. 마우스로 캐릭터의 방향을 바꾸기 위해서는 마우스 좌우 움직임이 Yaw 각도를 결정하고, 앞뒤 움직임이 Pitch 각도를 결정한다. (그림 3)은 게임 네비게이션 알고리즘을 설명한다.

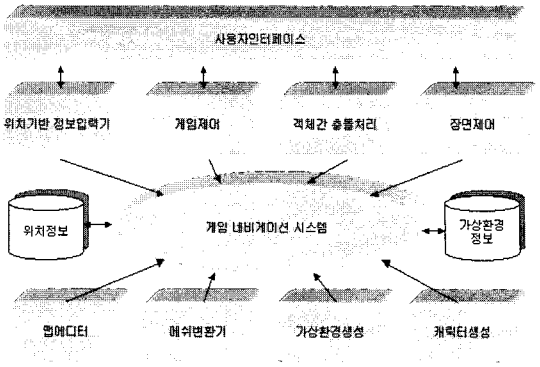
게임 네비게이션 진행 중에 발생하는 물체간의 충돌 처리를 위해 충돌 검사 알고리즘이 필요하다. 이를 위해 본 연구에서는 옥트리(Octree)를 사용하였다. 그리고, 가상환경을 구성하는 모든 객체를 구성하는 삼각형 전체에 대해서 충돌을 검사하는 방법은 효율적이지 않으므로, 메쉬를 전부 포함하는AABB(Axis Aligned Bounding Box, 축에 정렬된 경

```
// 캐릭터 네비게이션 알고리즘
(1) 캐릭터의 위치, 방향, 속도 값을 정의한다.
(2) 카메라 변환도 캐릭터 변환에 따라 이동되도록 하는데 카메
라의 위치과 초점을 캐릭터의 방향과 위치에 따라 맞춘다.
(3) 사용자의 키보드 입력과 마우스에 의한 네비게이션 제어를
위한 키 값을 얻음.
(4) (1)번과 (2)번 값으로부터 다음 방법으로 캐릭터 네비게이션
을 실행시킴
   - [A]와 [D] 키 값이 들어오면 캐릭터를 좌우 이동시킨다.
   - [W]와 [S] 키 값이 들어오면 캐릭터를 전진 및 후진시킨다.
   - 마우스의 앞/뒤 변화값이 들어오면 캐릭터를 Pitch 변환시
     킨다.
   - 마우스의 좌/우 변화값이 들어오면 캐릭터를 Yaw 변환시킨다.
(5) (4)번 프로세스로 캐릭터의 위치 변환이 일어날 때 마다 충돌
처리 프로세스가 가동된다.
(5) (4)번 프로세스로 캐릭터의 위치 변환이 일어날 때마다 시맨
틱 정보의 유무를 확인하여 화면에 출력한다.
```

(그림 3) 게임 네비게이션 알고리즘



(그림 4) 분리축(Separated axis)



(그림 5) 전체 시스템 구성

계 상자로 가상 환경 좌표계의 XYZ축과 평행한 상자)를 정의하여 메쉬의 각 삼각형을 검사하는 대신에 이 경계 상자를 검사함으로서 속도를 향상시키도록 하였다[6].

상자와 삼각형의 충돌 검사를 하는 알고리즘으로는 분리축(Separated Axis)를 사용하여 박스와 삼각형이 교차하는지 검사한다. 분리축이란 두 프리미티브를 분리하는 축을 의미한다(그림 4). 예를 들어 두 프리미티브가 떨어져 있다면 어떠한 축에 투영 시켰을 때 축에 투영된 두 프리미티브가 겹치지 않는다. 이 축을 분리축이라 부르며, 분리축 이론을 사용하여 교차 검사하여 분리축을 찾는다. 만일 이런 분리축이 존재한다면 두 프리미티브는 교차하지 않는다고 판정이 되며, 어떠한 분리축도 발견되지 않으면 두 프리미티브는 교차한다고 판정이 된다. 충돌 반응 처리에 있어서는, 충돌이 발생했다고 판정이 되면 캐릭터가 더 이상 그 방향으로 이동을 하지 않도록 하였다.

5. 게임 네비게이션 시스템 구현

본 시스템 구현을 위한 프로그래밍에는 비주얼C++과 DirectX 9를 사용하였고 배경화면과 캐릭터의 기하 데이터 제작에는 3DS MAX를 사용하였다. (그림 5)는 본 연구에서 개발된 전체 시스템 구성도를 나타낸다.

로고 화면, 인트로 동영상, 메인 메뉴, 게임진행 화면 등 여러 게임 장면들은 서로 독립적이기 때문에 장면별로 관리되는 로직이 필요하다. 이러한 장면들은 하나의 장면에서 다른 장면으로 전환되기도 하고, 장면이 시작할 때 처리해 주어야 할 작업과 장면이 끝날 때 처리해 주어야 할 작업이 존재하게 된다. 이러한 장면의 생명주기를 하나의 프레임워크에서 관리하게 된다.

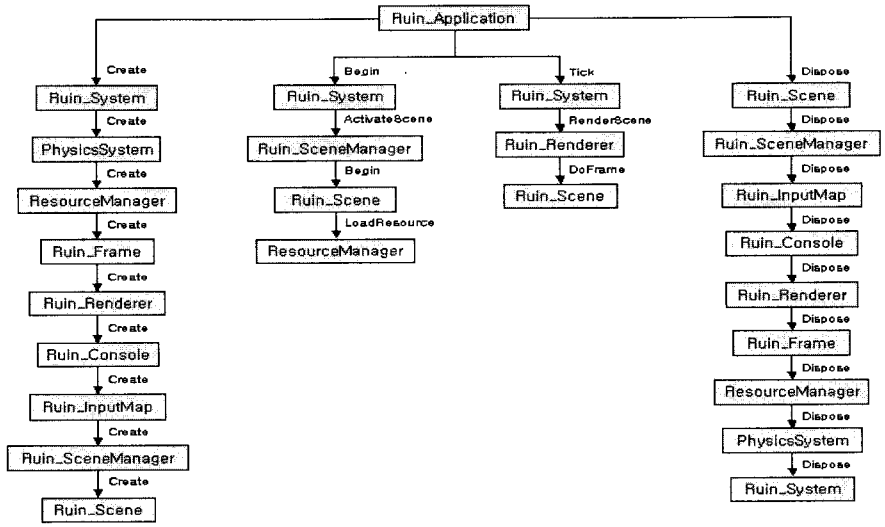
본 연구에서의 장면 관리 프레임워크는 SceneManager와 Scene의 2개 클래스로 구성되어 있다. 이것은 Scene 클래스를 상속받아 5개의 가상 함수 Initialize, Dispose, Begin, End, DoFrame을 재정의하여 SceneManager에 등록한다. 이와 같이 SceneManager가 생명 주기에 따라 함수를 호출하여 장면을 관리하도록 하였다. SceneManager는 등록된 장면들을 정수형의 아이디로 구분하며 특정 장면을 활성화하거나 제거할 때 이 아이디를 사용한다.

장면에는 활성화되어 있지 않은 장면과 활성화되어 있는 장면이 있다. 하나의 SceneManager에는 여러 개의 장면이 등록될 수 있지만 활성화될 수 있는 장면은 하나이다. 따라서 현재 활성화되어 있는 장면이 있는 상태에서 다른 장면을 활성화하려고 한다면 현재 활성화되어 있는 장면은 비활성화된 장면으로 변하게 된다.

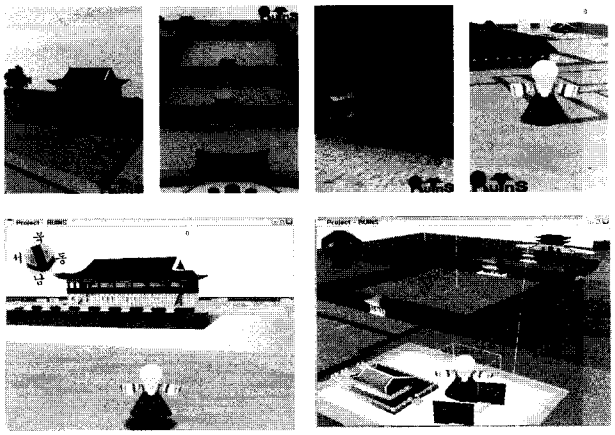
게임 실행 도중에 게임의 일부 작업을 수행하거나 속성을 변경하기 위해 하나의 클래스가 수행하는 Facade 패턴을 사용하여 구현하였다[12]. 사용자가 이 클래스에 접근할 수 있도록 콘솔 라이브러리를 사용하여 사용자가 콘솔 명령어로

〈표 1〉 게임 프로그래밍 클래스

클래스 종류	설 명	시스템구성
Ruin_Application	프로그램의 시작점	게임시스템
Ruin_System	게임의 여러 기능을 제공하는 시스템들을 전체적으로 관리	게임시스템, 게임제어
Ruin_PhysicsSystem	캐릭터와 주변 환경 사이의 상호 작용을 계산. 충돌 검사와 충돌 반응 처리 포함.	충돌처리
Ruin_Frame	Win32 API를 이용하여 윈도우를 생성하고 관리하는 클래스	게임시스템
Ruin_Renderer	렌더링에 사용되는 Direct3D 장치 관리	게임시스템
Ruin_Console	게임 설정을 조절할 수 있는 게임 내 콘솔 모드 관리	사용자인터페이스, 게임콘솔
Ruin_InputMap	DirectInput을 사용하여 사용자의 입력을 받아들임	맵에디터, 메쉬변환 위치기반정보입력기 가상환경생성, 캐릭터생성
Ruin_SceneManager	게임내 각각의 장면을 관리	장면제어
Ruin_Scene	게임내 각각의 장면을 정의하는 추상클래스. 실제 게임을 구현할 때 이 클래스를 상속받아 게임 내용을 구현	장면제어



(그림 6) 게임 네비게이션 시스템 구성 알고리즘의 다이어그램



(그림 7) 게임 네비게이션 실행 장면

게임 장면을 제어할 수 있도록 구현하였다.

다시 말해 콘솔 명령어로 게임의 여러 제어 기능을 실행시키도록 구성하는 사용자 인터페이스를 제공하였다. 기능적인 면에서 본다면 게임을 실행하는 도중에 기능을 사용자가 명령어를 통해서 실행할 수 있다는 장점이 있고, 시스템 관점에서 본다면 게임에 사용되는 기능을 일괄적으로 실행할 수 있는 창구 역할을 한다. 게임에 관한 여러 설정을 프로그램 내에서 정의하지 않고, 외부 배치 파일로 따로 빼놓음으로서 설정을 좀 더 유연하게 바꿀 수 있다.

게임 실행 제어를 위한 콘솔 명령어의 종류에는 콘솔 모드 설정 및 종료, 렌더링 모드 설정, 게임 중력 크기 설정, 환경 정보 파일 읽기, 카메라 설정, 뷰 속도 설정, 카메라 정보 저장, 카메라 정보 읽기, 명령어 파일 읽기, 카메라 정보 삭제 등이 포함된다. <표 1>은 본 게임 시스템에서 개발된 클래스 종류들이다. (그림 6)은 게임 네비게이션 시스템 전체 알고리즘의 구성도를 보여준다. (그림 7)은 게임 실행 장면들을 보여준다.

본 연구에서 개발한 게임 네비게이션 시스템은 기존 시스템과는 달리 환경 정보입력기를 제공하여 가상환경 내의 실세계에 해당하는 특정 위치에 텍스트 정보를 입력할 수 있도록 하였다. 이와 함께, 캐릭터 움직임을 네비게이션 중에 제어할 수 있도록 콘솔 명령어를 제공하고 있다.

6. 결 론

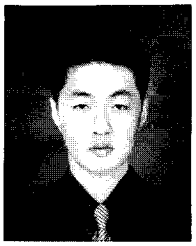
본 논문에서는 3차원 게임 개발에 있어서의 시맨틱 가상환경 생성과 네비게이션 제어를 위한 게임 프레임워크 구현에 대해 기술하였다. 이를 위해 3차원 가상환경의 생성, 시맨틱 가상환경 정보 입력, 게임 네비게이션 제어, 및 콘솔 명령을 포함하는 사용자 인터페이스 등이 개발되었다.

본 시스템은 기존 시스템과는 달리 두 가지 중요한 기능을 포함하고 있다. 하나는 게임 장면 생성에 있어서 게임 진행 장면 뿐 아니라 게임 장면의 대화형 편집을 제공하는 콘솔 명령 인터페이스를 제공한 것이다. 이 기능으로 게임을 진행하면서 장면의 저장, 변환, 수정 등이 실시간으로 이루어질 수 있도록 구성하였다. 다른 하나는 시맨틱 가상환경의 구현으로서 가상 환경 정보 입력기를 제공하여 실제 환경의 정보를 가상환경에 직접 실시간 입력 가능하도록 구성한 점이다. 이러한 시맨틱 가상환경 구현으로 실세계 위치 기반의 동적 정보 입력을 가능하게 한 점에서 향후 실세계 유비쿼터스 환경에서의 적용을 고려할 수 있다.

참 고 문 헌

[1] "Lars Bishop, Dave Eberly, Turner Whitted, Mark Finch, Michael Shantz, Designing a PC Game Engine", IEEE CG&A, Vol. 18, No. 1, 1998.

[2] Frank D. Luna, DirectX 9를 이용한 3D 게임 프로그래밍 입문, 정보문화사, 2004.
[3] Marc Erich Latoschik, Peter Biermann, Ipke Wachsmuth, "High-Level Semantics Representation for Intelligent Simulative Environments" Proceedings of IEEE Virtual Reality 2005, pp.283-284, 2005.
[4] Greg Snook저, C++와 DirectX9를 이용한 실시간 3D 지형 엔진, 정보문화사, 2004.
[5] Tomas Akenine-MollerEric Hanies 공저, 신병석, 오경수 공역, Real-Time Rendering 2판, 정보문화사, 2003.
[6] Akenine-Möller, Tomas, "Fast 3D Triangle-Box Overlap Testing," Journal of Graphics Tools, Vol. 6, No. 1, pp.29-33, 2001.
[7] K. Kanev, S. Kimura, "Integrating Dynamic Full-Body Motion Devices in Interactive 3D Entertainment", IEEE CG&A, pp.76-86, July 2002.
[8] Sung-Jin Kim, Falko Kuester, K. H. (Kane) Kim, "A Global Timestamp-Based Scalable Framework for Multi-Player Online Games", IEEE Fourth International Symposium on Multimedia Software Engineering (MSE'02), pp.2, December 2002.
[9] C. Faisstnauer, W. Purgathofer, M. Gervautz, J.-D. Gascuel, "Construction of an Open Geometry Server for Client-Server Virtual Environments", Proceedings of Virtual Reality 2001 Conference (VR'01), pp.105-114, 2001.
[10] J. Purbrick and C. Greenhalgh, "An Extensible Event-based Infrastructure for Networked Virtual Worlds", Proceedings of Virtual Reality, IEEE, pp.15-21, 2002.
[11] S. Srinivasan, "Design Patterns in Object-Oriented Frameworks" IEEE Computer, Vol. 32, No. 2, pp.24-32, February 1999.
[12] Marc Erich Latoschik, Peter Biermann, Ipke Wachsmuth. "High-Level Semantics Representation for Intelligent Simulative Environments," Proceedings of Virtual Reality 2005, pp.283-284, 2005.

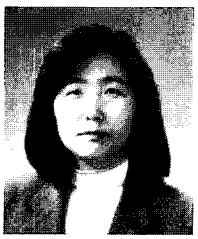


장 현 덕

e-mail : biggreat81@hotmail.com
2005년 2월 수원대학교 컴퓨터과학과 졸업
2007년 8월 인하대학교 대학원 정보공학과 졸업
관심분야 : 컴퓨터 그래픽스, 게임 엔진 제작

이 재 문

e-mail : breaklee80@estsoft.com
1999년 2월 수원대학교 컴퓨터학과 입학
2006년 2월 수원대학교 컴퓨터 학과 졸업
2007년 현재 ESTsoft 서비스 개발팀 재직 중
관심분야 : Enterprize solution, 3D data manipulation, XML web Service, RIA



이 명 원

e-mail : mwlee@suwon.ac.kr
1981년 서울대학교 졸업(학사)
1984년 서울대학교 대학원 계산통계학과 전산전공 졸업(석사)
1990년 일본 동경대학(The U. of Tokyo) 대학원 정보과학과 졸업(박사)
1990년~1993년 일본 Kubota Corp. 및 동경대학(The U. of Tokyo) 연구원
1993년~1996년 KT 멀티미디어연구소 선임연구원
1996년~현 재 수원대학교 컴퓨터학과 및 인터넷정보공학과 부교수
관심분야 : 컴퓨터그래픽스, 가상현실, 멀티미디어