게임 맵 디자인을 위한 협업 가상 환경에서의 동시성 제어 및 공동 편집 방법※

박성준[*1], 이 준[**], 임민규[**], 김지인[**]
호서대학교 게임공학과[*], 건국대학교 HCI 연구실[**]
sjpark@sjpark@hoseo.edu, {junlee, mlim, jnkm}@konkuk.ac.kr

A Concurrency Control and a Collaborative Editing Mechanism in a Collaborative Virtual Environment for Designing a Game Map

SungJun Park[*1], Jun Lee[**], MinGyu Lim[**], Jee-In Kim[**]
Game Engineering, Hoseo University, HCI Lab, Konkuk University

요 약

게임 레벨 디자인은 재미있는 게임 플레이를 위하여 매우 중요한 게임 제작 요소 중의 하나이며, 게임의 플레이가 이루어지는 공간의 구성 및 해당 맵의 제작, 세부적인 장치, 객체와 물체의 설정 및 배치, 배경 설정 및 이벤트 연출 등을 설계하는 과정이다. 게임 맵 디자인 과정은 다수의 3D 객체를 공간 속에 배치하고, 지속적인 평가, 수정 및 보완 과정을 통하여 게임 공간의 성능을 개선하여야 하므로, 그 과정에서 비용과 시간을 많이 사용하게 된다. 하지만 기존의 게임 제작 환경에서는 여러 개발자들이 공동으로 작업을 하게 되므로, 게임 공간 속의 객체 및 작업들에 대한 동시성 제어가 어려워서, 전반적인 작업의 일관성이 유지가 되지 못하는 문제가 발생하고 있으며, 또한 공동 편집 과정이 복잡해지면 공동 작업의 품질이 하락하는 문제가 발생하고 있다. 본 논문에서는 이러한 문제를 해결하기 위해 게임 맵 디자인을 위한 협업 가상 환경에서의 동시성 제어 및 공동 편집 방법을 제안한다. 제안된 시스템은 계층 구조 기반의 객체들을 사용하고, 각 객체별 동시성 제어 기법을 제공한다. 또한 공동 편집 작업을 원활하게 수행하기에 필요한 소유권 관리 기반 Undo/Redo 메커니즘을 제공한다. 실험 결과 본 논문에서 제안한 시스템을 사용 하는 경우에 충돌의 횟수가 감소하며, 다른 개발자들이 작업을 취소하여 재수정을 하는 등의 불필요한 작업 횟수가 감소하는 결과를 가져 왔다.

ABSTRACT

Game level design is a collaborative work to create a virtual world for a computer game including maps, agents, monsters, objects, players and events based on predefined its game scenario. It is a promising collaborative design application. The game level design generally requires much time and cost, as the size of its target game space becomes huge. However, traditional game level design tools do not provide concurrency control mechanisms among multiple participating game designers. They do not provide consistency of undo and redo mechanisms for erroneous collaborative tasks during iterative modifications and updates of collaborative tasks among multiple designers. In this paper, we propose a concurrency control and a collaborative editing mechanism to enhance productivity of the collaborative game level design. The proposed system provides hierarchical structures of shared objects and a concurrency control mechanism for each object. The proposed system also provides a consistent undo and redo mechanism to enhance modifications and updates on intermediate results of the level design procedures.

Keywords : Concurrency Control, Collaborative Editing, Level Design of Game Map, Collaborative Virtual Environment

접수일자 : 2011년 07월 08일 일차수정 : 2011년 08월 05일 심사완료 : 2011년 08월 12일
교신저자(Corresponding Author) : 김지인, 제1저자 : 박성준
※ 이 논문은 2010학년도 건국대학교의 연구년 교원 지원에 의하여 연구되었음.

1. 서 론

게임 맵 레벨 디자인이란 기획자가 창조한 게임 내용을 가상현실 환경 속에 해당되는 맵과 건물들 및 캐릭터들로 실체화 시키는 과정으로, 기획자, 원화가, 아티스트, 레벨 디자이너 등과 같이 여러 개발자들이 공동으로 협력해서 해결해야 하는 작업이다. 이러한 게임 레벨 디자인의 목표는 궁극적으로 게임의 재미를 제공 하는 것으로, 게임 제작의 성공 여부를 결정하는 매우 중요한 요소 중의 하나이다[1,2].

이러한 게임 레벨 디자인 제작 과정에는 일반적으로 많은 시간과 비용이 필요로 하며, 특히 규모가 큰 온라인 게임 같은 경우에는 천문학적인 제작비용과 시간이 소요된다. 따라서 이러한 시간과 비용을 효과적으로 줄이기 위해서는 기존의 오프라인 형태의 공동 작업의 과정을 온라인화 하여 협업 가상환경을 적용한 공동 작업 환경의 형태로 제공할 수 있다. 이러한 게임 레벨 디자인을 공동으로 적용 하는 연구들은 아직 초창기 단계라고 할 수 있다[3].

한편, 협업 가상 환경(Collaborative Virtual Environment) 은 다양한 작업들을 3차원 가상현실 공간에서 사용자들이 직관적인 상호작용을 사용하여 사용자간의 정보 및 경험을 공유하고 이를 통해 공동 작업을 수행하는 환경이다. 이러한 특징을 바탕으로 협업 가상환경은 온라인 게임, 군사 훈련, 의료 서비스, 과학 시뮬레이션 및 제품 설계 등의 다양한 분야에 적용이 되고 있다[4].

협업 가상 환경 속에서는 네트워크 환경을 이용하여 공동 작업에 참여하는 사용자들이 공유하는 객체에 대한 데이터 속성을 변경하고, 변경되는 속성을 모니터링 하며 결과물을 생성하고 수정 및 보완한다. 하지만 한 공유 객체에 대한 공동 작업을 동시에 시도하는 경우에 공유 객체의 속성 값을 어떤 사용자가 자신이 의도한 대로 임의로 바꾸게 되면, 다른 사용자가 피해를 보는 충돌 현상이 발생 하게 된다. 따라서 공유 객체의 속성 값을 어떤 사용자가 마음대로 변경하여, 다른 사용자들이 원하지 않거나 예상하지 못한 값으로 변경을 방지하고, 공유 객체의 속성 값의 일관성을 유지하기 위한 동시성 제어메커니즘이 필요하게 된다[5, 6]. 가장 일반적으로 많이 사용 되는 동시성 제어 방법은 여러 사용자가 공유 객체에 대한 동시 작업에 대한 접근을 시도하게 될 때 공유 객체의 소유권을 가진 사용자만이 공유 객체에 대한 작업을 수행할 수 있도록 허락해 주는 비관적 소유권 관리 방법(Pessimistic Concurrency Control Mechanism)이다. 이러한 동시성 제어 메커니즘은 협업 가상 환경에서 작동되는 수많은 애플리케이션들 중에서도, 공유 객체의 속성이 중요한 의미를 가지는 협업 디자인, 모델링 및 시뮬레이션 분야에서 유용하게 적용 될 수 있으며, 그 이외에도 관련된 여러 가지의 응용 분야에서 활발하게 연구가 되고 있다[7,13].

하지만 비관적 동시성 제어 방법을 사용 하더라도, 협업 가상환경에서 공유 객체들에 개한 공동 편집 작업을 수행하면서, 이전 작업에 대한 Undo 및 Redo 등 편집 작업을 수행하는 경우에는, 일관성 유지 문제가 발생할 수 있다. 예를 들어, 현재 공유 객체의 소유권을 가진 사용자가 해당 공유 객체의 이전 소유권자가 수행한 작업을 취소하게 되어 버리는 문제가 발생할 수 있다[14,15]. 이러한 일관성 유지에 관한 문제들이 자주 발생하는 경우에는 협업 가상환경에서 목표로 세우고 있는 공유 객체에 대한 공동 작업을 통한 생산성 향상을 달성하기가 어렵다.

본 논문에서는 이러한 문제를 해결하기 위해서 협업 가상 환경에서 게임 레벨디자인을 효율적으로 수행하기 위한 동시성 제어 기법 및 공동 편집 방법을 제안 한다. 제안한 시스템은 여러 개발자들의 공동 작업 환경을 지원해주면서 일관성 유지를 위한 비관적 소유권 관리 방식의 동시성 제어 기법을 제공한다. 여기에 객체의 소유권자가 공동 작업을 취소하기 위하여 Undo 작업을 수행 하는 경우에 이전 소유권자의 작업을 취소하게 되는 경우에

는 이전 소유자의 허락을 받아야 취소를 수행할 수 있게 허락하고, Redo의 경우에는 Undo가 수행된 경우에서 Redo를 수행해주는 공동 편집 방법을 제공한다.

본 논문에서는 제안하는 협업 가상 환경에서의 동시성 제어 기법 및 공동 편집 방법을 실제로 적용한 게임 레벨 디자인 시스템을 개발 하였으며, 해당 시스템에 대한 성능 평가 및 고찰을 위한 실험을 수행 하였다. 실험 결과, 공동 작업에서 발생하게 되는 충돌 문제 및 공동 편집 작업에서 발생하게 되는 추가적인 작업을 줄이게 되었으며, 이를 통하여 본 논문에서 제안한 시스템이 협업 가상 환경에서 게임 맵 레벨 디자인의 일관성 유지에 기여 할 수 있음을 알 수 있었다.

2. 관련 연구

2.1 공동 작업 환경에서의 동시성 제어

협업 가상 환경에서 공유 객체에 대한 동기화 및 동시성 제어를 위해 하나의 공유 객체를 세부적으로 잘게 나누거나 인터랙션 및 속성에 따라 객체의 소유권을 구분하는 연구들이 꾸준히 수행되어 왔다. 이러한 관련 연구들에 대해서 소개를 한다.

동시성 제어 기법은 크게 세 가지로 분류된다. 첫째, 낙관적 (Optimistic) 기법은 사용자들이 공유 객체에 대한 접근 및 수정을 기본적으로 허용을 하되, 충돌이 발생하는 경우에 가장 먼저 공유 객체에 접근을 한 사용자의 작업만을 인정 하고, 다른 사용자들의 작업은 Undo를 통하여 취소를 하는 방법 이다[5]. 하지만 Undo 과정을 통해 공유 객체에 대한 소유권을 신청한 사용자들 중 소유권을 가지는 한 명의 사용자를 제외한 다른 사용자들은 자신들이 수행했던 작업이 Undo 과정에 의해 취소되고 없어져 버리기 때문에 매우 놀라게 되는 일이 발생하게 된다[6]. 한편, CAD 분야에서는 낙관적 방법에 대한 다른 연구로, CAD 작업을 사용하고자 할 때, 작업들을 논리적으로 처리해서 객체들의 충돌이 발생하는 경우에 논리적으로 결과들을 정렬하여 제공해 줌으로써 작업들의 수행을 모두 허용 해주는 non-locking 기법을 지원하고 상호 관계에 따른 정리를 수행하는 시스템이 제안되었다[7]. 하지만 이 시스템에서 제안한 방법은 미리 정의된 범위 안에서만 작업들의 non-locking이 이루어지게 된다. 즉, 사용자들이 미리 정의 되지 않은 다른 작업을 수행 하는 경우에는 논리적인 결과의 정렬이 이루어 지지 않기 때문에 여전히 충돌이 발생하게 된다.

둘째, 비관적 (Pessimistic) 동시성 제어 기법에서는 공유 객체에 접근하기 위해서는 먼저 객체에 대한 소유권을 얻어야 하고, 소유권을 얻은 사용자만이 공유 객체에 대한 접근 및 작업을 진행할 수 있다. 이러한 비관적 방식 협업 작업에 참여하는 사용자들이 명확하게 소유권을 설정하고 이에 따라 공동 작업을 관리할 수 있다는 점에서 DIVE[8]와 같은 초기 협업 시스템부터 널리 사용 되고 있다. Li et al.은[9] 협업 디자인 시스템에서 비관적 방식을 사용하여 객체에 대한 소유권을 token으로 처리하며 한 번에 한 사람의 사용자만이 공유 객체의 속성을 수정할 수 있는 시스템을 제안 하였다. 하지만 이러한 시스템들을 사용하는 경우에는 객체의 소유권을 가진 사용자만이 공유 객체에 대한 작업을 수행할 수 있고, 다른 사용자들은 해당 공유 객체에 대해서 모니터링을 제외한 다른 공동 작업을 하기가 어렵다는 단점을 가지고 있다.

셋째, 예측(Predictive) 기반 동시성 제어 기법으로써 협업 가상환경에서 참여 사용자들의 공유 객체에 대한 접근 및 작업 방법을 미리 예측하여 가장 확률이 높은 사용자에게 미리 소유권을 주는 알고리즘으로 PaRADAE[11]와 같은 시스템에서 제안이 되었다. 또 다른 예측 기법으로, 아바타를 통하여 공동 작업을 하는 가상 환경에서 공유 객체에 대한 소유권을 예측하기 위해서 공유 객체의 반지름과 사용자들과의 거리, 방향을 계산하여 해당 공유 객체를 소유할 확률이 가장 높은 사용자

에게 소유권을 제공해주는 연구가 제안되었으며[11] 이를 개선하여 아바타들의 이동 속도까지 고려하여 공유객체에 대한 소유권을 예측하는 연구들이 이루어졌다[12]. 하지만 이러한 예측 방법은 예측이 잘못되어 틀린 경우, 소유권 재설정에 소요되는 오버헤드가 들며, 이런 특성 때문에 정밀한 작업이 필요한 공동 디자인 시스템 같은 협업 환경에는 적절하지 않은 방법 이다. 따라서 일반적으로 협업 가상 환경에서 정밀한 모델링 및 시뮬레이션이 필요한 장면은 비관적 접근 방법을 사용한 동시성 제어 메커니즘을 주로 사용한다[7,13].

한편, 공동 작업 환경에서는 사용자들이 작업을 하는 도중에 실수, 시스템 에러들 및 자신들의 작업 결과가 마음에 들지 않는 경우에는 해당 작업들을 취소할 수 있어야 한다[14,15]. 하지만 공유 객체에 대한 작업 결과들을 취소하는 경우에 가장 먼저 다른 사용자와의 동시성 작업에 대한 일관성 문제가 발생하게 된다. 또한, 비관적 방식의 소유권 제어 방식을 사용한 공동 편집 작업 도중에 Undo 기능을 지원하더라도 해당 공유 객체에 대한 이전 소유권을 가진 사용자들의 작업들을 취소하게 되는 문제가 발생 하게 된다. 이러한 경우에 기존의 연구들에서는 일반적으로 Undo를 하거나 Redo를 하는 과정에서 이와 같은 문제가 발생하게 되면 공동 작업 공간에 복사본을 생성하는 방법으로 문제를 해결하였다. 하지만 일반적으로 공동 작업 공간에 대해서 한 공유 객체에 대한 다양한 복사본이 나오게 되는 경우에는 다시 복사본과 원래의 원본 데이터 간의 일관성을 유지해야 하는 문제가 발생한다.

2.2 레벨 디자인에서의 공동 편집 작업

레벨 디자인 과정은 게임 개발 과정에서 많은 비용이 소요 되는 과정으로 재미있는 게임을 만들기 위해서 여러 명의 기획자 및 레벨 디자이너들이 공동으로 맵을 만들고 이에 관련된 레벨을 디자인 하게 된다. 이러한 레벨 디자인의 중요성에 따라 게임이 흥행에 성공 및 실패를 결정 할 정도로 게임 제작 과정에서 매우 중요한 요소라고 볼 수 있다[1,2].

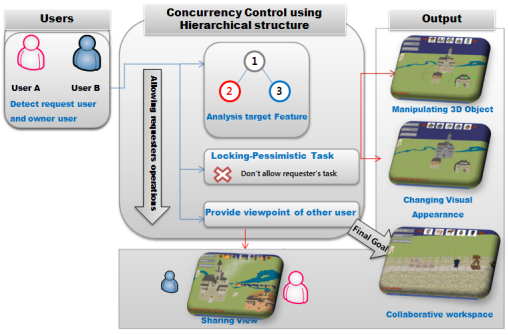
하지만 이러한 레벨 디자인 과정은 기본적으로 많은 시간과 비용을 필요로 하게 되므로 공동 작업이 필요한 상황이다. 기존의 대부분의 레벨 디자인 과정은 오프라인 형태에서 개인적으로 작업한 부분적인 결과들을 결합하여 작업을 하게 된다. 하지만 이러한 경우 사람들이 각자 작업을 한 결과물에 대한 디자인의 통일성 및 일관성을 유지하기가 어렵다는 단점을 가지고 있으며, 또한 개인적으로도 레벨 디자인 과정을 하는 과정에 시간과 비용이 많이 소모된다는 단점이 있다.

이러한 문제점을 해결하기 위해서는 협업 가상 환경을 적용하여 공동 작업 환경을 지원해야 한다. 대표적인 사례로 Hero Engine에서는 인터넷을 사용하여 다수의 개발자가 레벨 디자인을 할 수 있는 시스템을 제공하고 있다[3]. 하지만, Hero Engine의 경우에는 기본적인 협업 가상환경만을 제공해 주고 있기 때문에 공동 작업을 하는 객체에 대한 조작이 현재 어느 개발자에 의해서 이루어지고 있는지에 대한 UI의 구성이 제한되어 있고, 다른 개발자들의 공동 작업 과정을 알려 줄 수 있는 상호작용이 부족하다. 다른 고려사항으로는 한 객체를 여러 개발자가 조작 하는 경우에 대한 동시성 제어 방법을 제공하고 있지 않으며, Undo 및 Redo 기능 등을 사용한 공동 편집 작업의 경우도 이전에 수행 하였던 소유권자의 작업을 허락 없이 취소를 해버리는 문제가 발생하게 된다.

다른 사례로 Epic사의 Unreal 엔진의 경우에는 엔진의 기능이 매우 막강하며 다양한 편집 기능을 제공해주고 있으며, 대규모 온라인 게임에서의 공동 작업이 가능한 Atlas Technology라는 통합 프레임워크를 제공하고 있다. 이 프레임워크를 통해서 게임 개발자들은 레벨 디자인 과정뿐만이 아니라, 리소스 관리, 애니메이션 편집 등의 공동 작업을 할 수 있게 지원해 준다. 하지만 Atlas Technology는 레벨 디자인 과정에서 생성되는 스크립트에 대한 버전 컨트롤 기능을 제공하고 있기

때문에, 본 논문에서 언급한 것과 같이 세밀한 레벨 디자인 공동 작업을 여러 개발자들이 동시에 수행 할 때, 발생할 수 있는 충돌 현상에 대한 고려는 따로 되어 있지 않은 상황이다[16]. CryTek의 CryEngine의 경우도 기본적으로 공동 작업을 위한 스크립트에 대한 버전 컨트롤 관리 기능에 더불어 Live Create라는 기술을 통하여 PC, XBOX360 및 플레이스테이션 3에서 디자인 과정 및 플레이를 경험할 수 있는 협업 가상 환경을 제공 하고 있으나[17], 본격적인 레벨 디자인 과정에서 발생할 수 있는 일관성 유지 문제에 대한 고려는 따로 지원 하고 있지 않다. 따라서 이를 실제로 게임 개발에 적용하기 위해서는 XLGAMES의 아키에이지와 같이 CryEngine 엔진의 기술을 응용하여 독자적인 공동 레벨 디자인 환경을 구축하여 사용해야 한다[18]. 아키에이지 게임 사례에서는 레벨 디자인 과정 중 맵의 터레인(Terrain, 지형)을 제작하는 과정에 대해서 소유권에 따라 터레인의 높이 조절 및 텍스처 조절을 수행하는 작업을 구축 하였다.

3. 시스템 구성

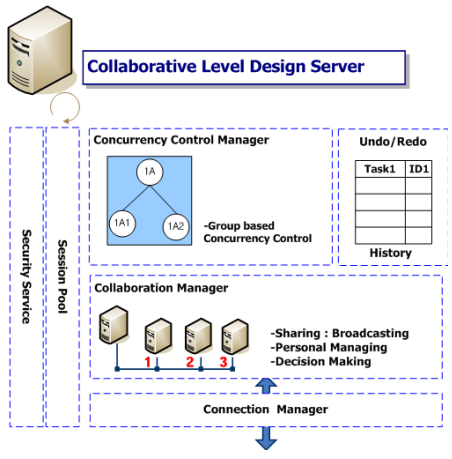


[그림 1] 협업 가상 환경 기반의 게임 레벨 디자인 시스템

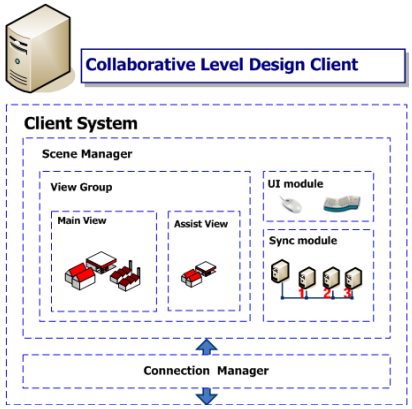
본 논문에서는 협업 가상 환경에서의 레벨 디자인 과정에서 발생하는 공유 객체에 대한 작업 과정에 따른 일관성 충돌 문제를 해결하면서, 공동 편집 작업 과정에서 발생할 수 있는 에러 값들의 결과에 대한 수정 시에 발생하는 Undo 및 Redo 문제를 해결할 수 있는 시스템 및 제어 방법을 제안한다. 위의 [그림 1]은 본 연구에서 제안하는 시스템의 구조이다.

제안하는 시스템의 특징은 협업 가상 환경에서의 레벨 디자인을 지원하면서, 여러 개발자들의 정밀한 조작 과정에 발생할 수 있는 일관성 충돌을 해결하기 위해 비관적 소유권 제어 방법을 사용한 동시성 제어 메커니즘을 사용한다. 또한 관련연구에서 제기된 비관적 소유권 방식을 사용 하는 경우의 다른 개발자들의 행동 참여 제한을 해결하기 위하여, 공유 객체들을 계층 구조로 구성할 수 있도록 지원함으로써 공유 객체에 대한 접근성을 높여주게 된다. 또한, 공동 작업에 대한 분류를 하여 일반적인 공동 작업에 대한 동시성 제어 기법을 제공하고, Undo 및 Redo를 사용한 공동 편집 작업에서이전 소유권을 가진 개발자들의 작업을 함부로 취소하지 않고 추가적인 허락을 통한 작업을 수행함으로써 작업의 전체적인 일관성 유지를 보다 향상시켜 줄 수 있는 시스템 이다.

3.1 서버 및 클라이언트의 구조 및 기능



[그림 2] 협업 레벨 디자인 서버의 시스템 구조

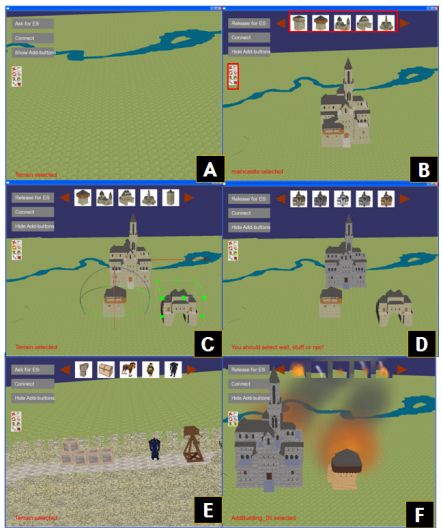


[그림 3] 협업 레벨 디자인 시스템의 클라이언트 구조

본 논문에서 제안하는 시스템은 협업 가상환경의 효율적인 관리를 위하여 서버 및 클라이언트 구조로 구성되어 있다. 다음의 [그림 2]는 본 논문에서 제안하는 시스템 중 서버의 모습이다. 기본적으로 협업 레벨 디자인 서버에서는 클라이언트와의 연결을 담당하는 Connection Manager를 통하여 클라이언트에서 개발자가 요청한 협업 작업을 분석하고 이에 맞게 처리하는 Collaboration Manager로 전달된다. 이후 해당 요청이 동시성 제어 작업인 경우에 이를 담당하는 Concurrency Control Manager를 통해 처리가 이루어지고, 공동 작업의 편집 작업의 경우에는 Undo/Redo Manager를 통해서 이루어지게 된다.

[그림 3]은 협업 게임 맵 레벨 디자인 클라이언트의 구조를 보여준다. 클라이언트에서는 기본적으로 맵 레벨 디자인을 처리하고 효과적인 렌더링을 하기 위해서 OpenSceneGraph 라이브러리를 사용한다. 이후 협업 작업을 사용하는 경우에는 서버와의 연결을 담당하는 Client Manager를 통하여 연결이 된다. 이후 개발자는 게임 맵의 레벨 디자인 작업을 Scene Manager를 통해 수행 하면서 공동 작업을 수행하게 된다.

개발자들은 다음의 [그림 4]와 같이 다양하고 직관적인 UI를 적용하여 해당 객체에 대한 공동 작업을 수행할 수 있다. 여기서, 공유 객체에 대한 3차원 객체 이동, 회전등의 변환을 할 때는 [그림 4]의 C와 같이 공유 객체에 나타낸 3D 조작 모델에 대해서 마우스를 사용한 클릭 및 드래깅(dragging)등의 상호작용들을 통하여 작업을 수행할 수 있다.



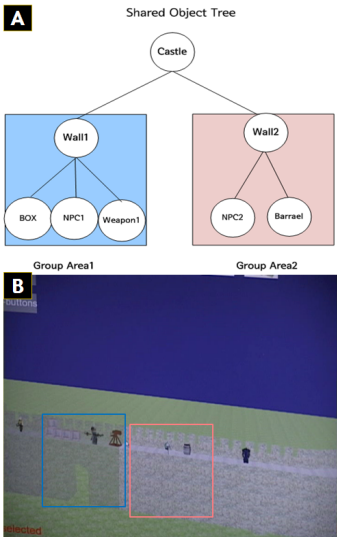
[그림 4] 맵 디자인 과정에서 직관적인 UI를 적용한 객체 조작 (A) 터레인 생성 (B) 3D 객체 추가 (C) 3D 객체에 대한 회전, 이동, 크기변환 조작 (D) 3D 객체에 대한 텍스처 변환 (E) 계층 구조 처리 (F) 파티클 시뮬레이션

3.2 공동 작업에서의 동시성 제어 기법

본 논문에서 제안하는 시스템을 사용하여 공동 레벨 디자인을 하는 경우, 먼저 개발자로부터 게임 레벨 디자인 과정에서 조작하고자 하는 객체에 대한 입력을 받게 된다. 이 객체들은 레벨 디자인 과정에서 상황에 따라 계층적으로 구성 될 수 있으며, 개별적으로 소유권한을 가지게 된다. 개발자들은 개별적인 객체에 대한 소유 권한을 신청한 뒤, 해당 객체에 대한 조작을 수행할 수 있다. 이때 개발자의 소유권한을 제어하기 위해서 비관적 접근 제어 방식을 적용하여[9] 해당 개발자가 해당 공유 객체에 정당한 소유권을 가지고 있는지를 판단한 뒤, 소유권이 없는 경우에는 개발자의 접근을 거부

하게 된다.

한편 공동 레벨 디자인에 참여하는 개발자들은 공유 객체들에 대한 소유권을 개별적으로 적용할 수도 있으며, 또한 계층적 구조로 이루어진 공유 객체들의 그룹을 전체로 처리하여 조작 할 수 있게 된다. 다음의 [그림 5]의 (A)는 여러 개의 공유 객체들을 하나의 계층 구조로 만든 예이며, [그림 5]의 (B)의 경우는 이를 실제로 클라이언트 화면에서 보는 경우이다. 이 경우 전체 그룹에 대한 소유권을 바탕으로 그룹에 속하는 모든 공유 객체들의 조작이 가능 하다.

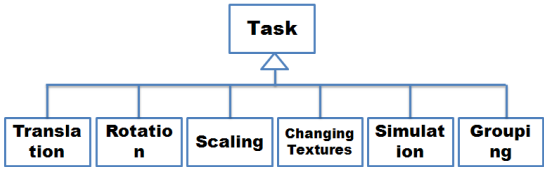


[그림 5] (A)계층 구조기반의 그룹으로 구성된 공유 객체들 (B) 클라이언트 화면에서 공유 객체들의 선택된 모습

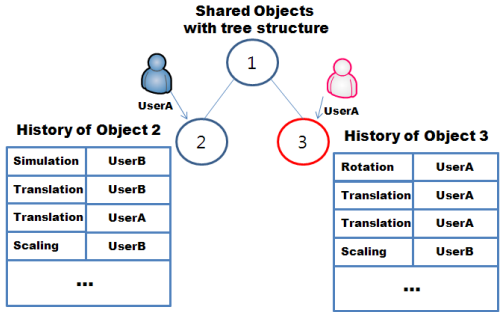
3.3 공유 작업에 따른 분류 및 공동 편집 방법

본 논문에서 제안한 시스템은 협업 가상 환경에서의 공유 객체에 대한 동시성 제어 기법을 제공하고, 수행된 작업에 대한 공동 편집 방법을 제공한다. 이를 위하여 개발자들의 공동 작업들은 다음의 [그림 6]처럼 공유 작업들에 대한 Command 패턴을 적용한다.

즉 공동 작업의 정보는 개발자의 요청에 따라 객체의 위치 정보를 조작하는 Translation, Rotation, Scaling으로 구분된다. 공동 작업 정보는 공유 객체의 외관을 결정 하는 Changing Textures로 처리된다. 여기서 Simulation은 공유 객체가 가지고 있는 연기 및 불꽃의 파티클 시뮬레이션(Particle Simulation) 등을 표현할 수 있다. 마지막으로 계층 구조구성을 위한 그룹화에 관련된 작업을 하는 Grouping 작업으로 분류된다. 이렇게 분류된 공동 작업 정보들은 개발자들이 공동 작업을 수행하게 될 때, 각 공유 객체 별로 다음의 [그림 7]과 같이 History 정보에 저장된다. 이때, 현재 공동 작업을 수행하는 개발자의 정보도 같이 저장된다.



[그림 6] 공동 작업에 대한 Command 패턴을 적용한 분류 방법

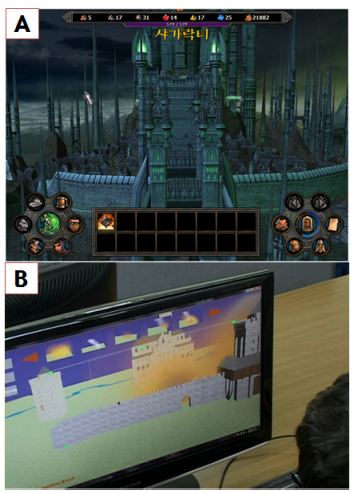


[그림 7] 공유 객체에 대한 공동 작업에 대한 History 정보 관리

이후, 해당 공유 객체에 대한 소유권을 가진 개발자가 해당 공유 객체에 대한 작업들에 대해서 Undo 및 Redo를 수행하게 되는 경우에 해당 작업을 History Table로부터 참조하여 Undo 및 Redo 작업을 수행하게 해준다. 이러한 상황에서 해당 공

유 객체에 대한 이전 소유권자가 수행한 작업에 대해서 취소를 수행하고자 하는 경우에는 이전 작업의 소유권자에게 해당 작업에 대한 취소 여부를 문의 하고, 해당 작업을 취소하게 된다. 이후 개발자가 Undo한 작업에 대해서 Redo 작업을 수행하고자 하는 경우에는 시스템에서 기본적으로 허락을 해주게 된다.

4. 실험 결과 및 고찰



[그림 8] (A) 실험에서 타겟 이미지로 제시된 Heroes of Might and Magic5의 성 이미지 (B) 레벨 디자인 과정에서 실험 참여자가 성벽을 구축 하고, 시뮬레이션 테스트를 하는 모습

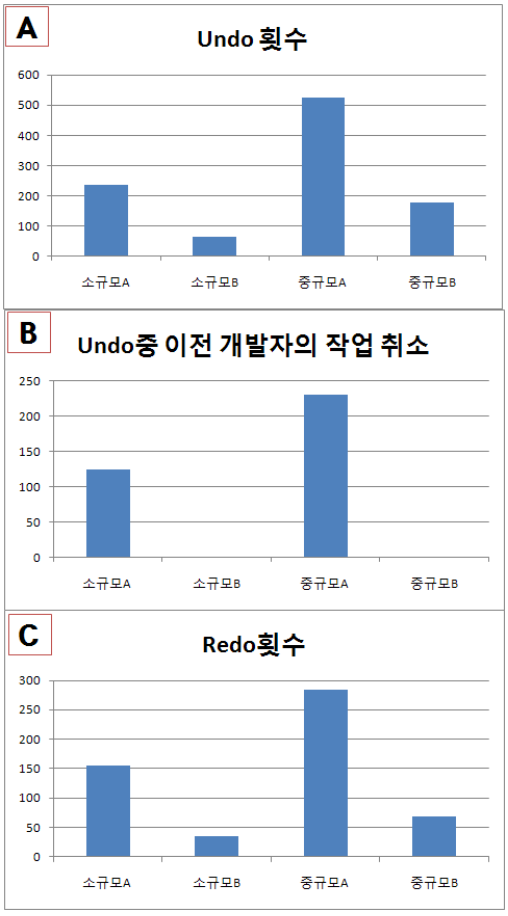
본 논문에서 제안한 방법의 성능평가를 위해서 본 논문에서 제안한 방법과 기존의 방법에 대한 비교 실험을 수행하였다. 실험은 각각의 방법들을 사용하여 레벨 디자인하는 경우에 걸린 시간, 총 작업 횟수, 충돌 횟수 및 Undo 및 Redo 수행 횟수와 비율을 측정하였다. 실험을 위하여 본 연구에서는 MMORPG 게임 환경에서 성 및 요새를 디자인 하는 과정을 실험모델로 설정하였다. 또한 본 실험에서는 레벨 디자인 과정에서 사용된 리소스들은 레벨 디자인 분야에서 각광 받고 있는 모듈 기반의 리소스 디자인 방법론[19]을 사용하였으며, 실험에 사용된 모듈들은 13개의 성과요새를 구축하는 건축물들, 5개의 요새 건물들, 4개의 NPC 구조물들 및 18개의 기타 구조물들로 구성된다. 실험 대상으로는 3D 객체 조작에 익숙하고, 레벨 디자인 경험이 있는 실험자들을 30명을 선별하였다. 본 논문에서 제안한 방법을 사용한 레벨 디자인 과정에서, 너무 소규모의 인원을 구성하여 실험을 하는 경우에는 공동 작업에서 충돌이 많이 발생하지 않고, 게임 개발 과정에서는 적용되기 어렵기 때문에, 소규모 개발인 경우를 5명으로 선정하였으며, 중규모 이상의 경우를 10명으로 선정하였다. 각 규모 그룹은 각각 기존의 방법을 사용하는 경우 및 본 논문에서 제안한 방법을 사용한 경우로 나누어서 구성되었다.

본 논문에 참여한 실험 대상은 본 논문에서 제안한 시스템에 대한 교육 과정을 거친 다음의 [그림 8]과 같이 실험에서 기획하고 만들고자 하는 Heroes of Might and Magic5(HOMM5) 게임의 이미지를 보여주고, 이와 유사한 느낌이 날 수 있는 레벨 디자인을 구성하는 실험을 진행하였다. 실험이 진행 되는 동안 참여자들은 의사소통을 통하여 서로간의 역할 분담을 하고 작업하였으나, 레벨 디자인 과정에 처음에 합의했던 부분이 맞지 않는 경우에는 다른 사용자들의 작업을 가져가거나 수정을 하는 등의 작업을 진행하였다. 실험의 완료는 실험 모델로 나온 22개의 모듈들의 복사본들을 배치하여 [그림 8](A)와 같이 본성, 교회, 병영, 감시타워들의 배치가 되어 있으며, 성벽들의 구축이 각각 앞문과 후문의 배치가 이루어지고, 성벽 위에 공성 무기들과 경비병들의 배치가 완성이 된 경우에 실험을 종료하였다. 다음의 [표 1]은 각 실험에서 소요된 실험의 완료 시간을 보여준다. 단, 여기서 A그룹은 기존의 방법을 사용한 경우이며, B그룹은 본 논문에서 제안한 방법을 사용한 경우이다.

[표 1] 공동 작업 실험에서 레벨 디자인 완료를 위해 걸린 시간

	최종 작업횟수	충돌 발생 횟수	걸린 시간
소규모A	1,215	481	1시간 13분
소규모B	526	240	47분
중규모A	3,213	1,135	2시간 20분
중규모B	1,578	896	1시간 5분

[표 1]의 결과와 같이 본 논문에서 제안한 방법을 사용한 소규모 팀B에서의 결과가 가장 좋게 나온 것을 알 수 있다. 이러한 이유는 실험 인원이 5명인 경우에는 실험 참여자간에 의사소통이 어느 정도 잘 이루어지고, 역할 분담이 이루어지면서 작업을 했기 때문에, 공동 작업에서의 충돌이 상대적으로 적게 발생하였고, 충돌이 발생한 경우에서의 대처 및 해결이 빠를 수 있었다. 하지만, 인원이 중규모 이상인 10명으로 넘어가면서, 역할 분담이 중복되고, 많은 인원들이 서로 의견을 주장하여 합의점을 찾기가 어렵고 또한, 서로 객체 조작을 하려고 하는 경쟁 상황이 발생해서, 오히려 소규모 그룹 보다 더욱 많은 공동 작업 횟수를 수행 하면서도 시간이 많이 걸리는 흥미로운 결과가 발생하였다. 또한 공동 작업 과정에서 충돌이 발생하는 경우에는 기존의 방법에서는 충돌이 발생을 시킨 사용자들의 작업 결과가 일치하지 않기 때문에 이후 사용자와의 의사 협약을 통해 다시 수정을 해야 하는 상황이 발생하게 되어 작업 횟수가 상대적으로 증가하였었다. 하지만 본 논문에서 제안한 시스템을 사용 하는 경우에는 충돌이 발생하였지만, 해당 객체에 대해서 먼저 작업을 하던 소유권자의 작업이 그대로 보호되고, 이후에 작업을 신청한 사용자들의 작업들을 블록킹을 하였기 때문에, 충돌이 발생하여 이 공유 객체의 값을 다시 수정하는 작업이 상대적으로 줄에 들게 되었다. 또한 해당 실험을 하는 과정에서 사용자들이 공동 작업 중에 작업을 취소 및 재취소를 하게 되는 비율을 측정해 보았다. 다음의 [그림 9]는 이러한 실험 그룹 간의 취소 및 재 취소 횟수를 보여준다.



[그림 9] (A) 실험에서 각 그룹 에서 발생한 Undo 횟수 (B) 실험에서 각 그룹 에서 발생한 Undo중 이전 개발자의 작업 취소 횟수 (C) 실험에서 각 그룹 에서 발생한 Redo 횟수

[그림 9]의 결과와 같이 본 논문에서 제안한 방법을 사용한 경우에, 사용자들의 공동 작업 과정에서 Undo를 사용 하는 비율이 15~30%임을 알 수 있다. 기존의 방법에서 Undo 횟수, Undo 중 이전 개발자의 작업 취소 및 Redo 횟수가 높게 나온 이유는 실험 중에 충돌이 발생한 경우, 기존의 방법은 충돌로 만들어진 공유 객체의 결과가 충돌에 참여한 사용자들의 의도와 모두 달라진다. 따라서 이 경우에 사용자들은 해당 작업을 오류라고 생각을 하게 되고 Undo를 수행하여 다시 작업을 수행하게 된다. 본 논문에서 제안한 방법을 사용 하는

경우에는 충돌이 발생하더라도, 가장 먼저 작업을 수행한 소유권자의 작업을 보장해 주기 때문에 상대적으로 Undo 횟수가 적게 나왔다. 또한 [그림 9]의 (B)에 나온 Undo 중 이전 개발자의 작업을 취소하는 상황도 3~40% 정도의 비율로 발생하게 되었다. 특히 역할 분담이 중첩되고, 작업 시간이 오래 중첩이 된 경우에 이러한 상황이 발생하였으며, 이 경우에는 해당 공유 객체에 대한 작업하였던 개발자의 항의를 받아서 다시 Redo를 수행하게 되는 횟수가 증가 되는 상황이 발생하였다. 하지만, 본 논문에서 제안한 방법을 사용 하는 경우에는 소유권 관리 기반의 History 정보 추적을 통한 허락 유무를 결정하였기 때문에 이러한 중복 될 수 있는 작업을 줄일 수 있었다.

실험 후에는 실험에 참여자들에 대한 인터뷰를 수행 하면서 협업 가상 환경에서 공동 레벨 디자인 작업을 하는 경우에 공유 객체들에 대한 공동 작업 과정에서 일관성 문제가 발생할 수 있다는 점을 공감하였다. 일반적으로 실험 참여자들은 실험 과정에서 이러한 문제를 줄이기 위해서 실험 과정에서 자연스럽게 역할 분담을 하였으나, 공유 객체들의 복잡성 증가 및 의견 일치 과정에서 공동 편집 작업에서의 일관성 충돌 현상이 발생 할 수밖에 없음을 인지하게 되었다. 사용자들은 공동 작업을 통해서 기존의 작업을 빠르게 향상할 수 있음에 공감 하면서도 본 논문에서 제안한 시스템과 같이, 공동 작업 중에서 동시성 제어 기법 등을 적용한 전체적인 일관성을 유지하는 작업이 필요하다는 의견을 주었다.

5. 결론 및 향후 연구

본 논문에서는 게임 레벨 디자인 작업의 성능 향상을 위한 협업 가상 환경을 적용한 게임 맵 레벨 디자인 시스템을 제안하였다. 제안하는 시스템은 다수의 개발자의 참여를 허용하고, 다수의 개발자들이 공동 작업 공간에서 계층화된 공유 객체들에 대한 비관적 동시성 제어 기법을 제공한다. 또한, 개발자들의 공동 작업들에 대한 Undo 및 Redo의 수요가 발생하는 경우에 이들에 대한 동시성 제어 기법을 지원하여, 공동 레벨 디자인에서 발생할 수 있는 공유 객체들의 일관성을 유지하고 전체적인 통일성을 유지할 수 있는 기본적인 구조를 제안하였다.

향후에는 보다 많은 개발자들이 공동 작업의 기회를 참여할 수 있는 기회를 제공하는 공동 작업 환경에서의 개발자 제약문제를 해결하고, 이를 바탕으로 공동 작업 개발자들이 동시에 작업을 하는 동기식 협업만이 아니라, 작업자들이 상대방의 작업 시간에 구애 받지 않고 서로가 편한 시간에 참여할 수 있어서 서로 다른 시간대에도 공동 작업을 수행할 수 있는 비동기식 협업을 통한 공동 작업 스키마를 개발하여 적용할 예정이다.

참고문헌

[1] Rob Pardo, “Making a Standard(and Trying to Stick to it!): Bizzard Design Philosophies”, GDC 2010.

[2] Phil Co, “Level design for games”, New Riders Games, 2006, pp 353.

[3] Hero Engine, www.heroengine.com

[4] Chris Joslin, Thomas Di Giacomo, and, Nadia Magnenat-Thalmann, “Collaborative virtual environments: from birth to standardization”, Communications Magazine 42(4) 2004, IEEE, 22-23.

[5] Un-Hae Sung, Jae-Heon Yang, Kwang-Yun Wohn, “Concurrency Control in CIAO”, Proceeding of the 1999 IEEE Virtual Reality Conference (VR’99), 22-28

[6] John M. Linebarger, G. Drew Kessler, “Concurrency Control Mechanisms for Closely Coupled Collaboration in Multithreaded Peer-to-Peer Virtual Environments”, Presence 13(3), 296-314, 2004.

[7] Min Tang, Shang-Ching Chou, and Jin-Xiang Dong, “Collaborative virtual environment for feature based modeling”,

Proceedings of the 2004 ACM SIGGRAPH international conference on Virtual Reality continuum and its applications in industry, 120-126.

[8] Olof Hasgand, "Interactive multiuser VEs in the DIVE system", IEEE Multimedia 3(1), 30-39, 1996.

[9] W.D. Li, S.K. Ong, J.Y.H. Fuh, Y.S. Wong, Y.Q. Lu, and A.Y.C Nee, "Feature-based design in a distributed and collaborative environment. Computer-Aidede Design", 36, 777-790, 2004.

[10] David J. Roberts, and Paul M. Sharkey, "Maximizing concurrency and scability in a consistent, causal, distributed Virtual Reality system, whilst minimizing the effect of network delays", Proceedings of the Sixth Workshop on Enabling Technologies Infrastructures for Collaborative Enterprise (WETICE'97), 161-16, 1997.

[11] Jeonghwa Yang, and Dongman Lee, "Scalable prediction based concurrency control for distributed virtual environments", Proceedings of the 2000 IEEE Virtual Reality Conference (VR 2000), 151 - 158, 2000.

[12] Eunhee Lee, Dongman Lee, Seunghyun Han, and, Soon J. Hyun, "Prediction-based concurrency control for a large scale networked virtual environment supporting various navigation speeds", ACM VRST 2001, 127-134.

[13] Jun Lee, PhamSy Quy, Jee-In Kim, Lin-Woo Kang, Anna Seo, HyungSeok Kim, "A Collaborative Virtual Reality Environment for Molecular Biology", ISUVR 2009, pp.68-71.

[14] David Chen and Chengzheng Sun, "Undoing any operation in collaborative graphics editing systems", In Proceedings of the 2001 International ACM SIGGROUP Conference on Supporting Group Work (GROUP '01) 2001, 197-206.

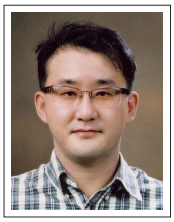
[15] Weiss, S., Urso, P., Molli, P., "Logoot-Undo: Distributed Collaborative Editing System on P2P Networks", IEEE Transactions on Parallel and Distributed Systems 21(8) 2010, 1162-1174.

[16] Unreal's Atlas Technology, www.epicgameschina.com/tech/tech-atlas_features.html

[17] CryEngine3, http://crytek.com/cryengine/cryengine3/overview

[18] 송재경, "FPS엔진 MMORPG에 적용하기", 2010 게임테크 기조연설 아키에이지편.

[19] Perry, L., "Modular Lavel and Component Design", Game Developer Magazine, 2002, pp 30-35.



박 성 준 (SungJun Park)

1997년 호서대학교 컴퓨터공학과(학사)
1999년 건국대학교 컴퓨터공학과(석사)
2005년 건국대학교 컴퓨터공학과(박사)
2006년-현재 호서대학교 게임공학과 조교수

관심분야 : 게임공학, 가상현실, HCI, Bioinformatics

이 준 (Jun Lee)

2004년 건국대학교 컴퓨터공학과(학사)
2006년 건국대학교 컴퓨터공학과(석사)
2006년-현재 건국대학교 신기술융합과 iIT 박사과정.

관심분야 : 협업 가상현실, 컴퓨터 그래픽스, HCI

임 민 규 (MinGyu Lim)

1998년 KASIT 전산학과(학사)
2000년 ICU 공학부(석사)
2006년 ICU 공학부(박사)
2006년-2008년 제네바대학 MIRALab 선임연구원
2009년-현재 건국대학교 조교수

관심분야 : 네트워크가상환경, 유비쿼터스컴퓨팅 시스템

김 지 인 (Jee-In Kim)

1982년 서울대학교 컴퓨터공학과(학사)
1984년 KAIST 전산학과(석사)
1993년 University of Pennsylvania 전산정보학 박사
1982년-1987년 금성통신 연구소
1993년-1995년 미국 CCCC 연구원
1995년-현재 건국대학교 교수

관심분야 : HCI, 가상현실