다양한 이동속도를 지원하는 대규모 네트웍 가상 환경을 위한 예측 기반 동시성 제어

이은희[0] 이동만 한승현 현순주
한국정보통신대학원대학교 공학부
{ehlee, dlee, dennis, shyun}@icu.ac.kr

Prediction-based Concurrency Control for A Large Scale Networked Virtual Environment Supporting Various Navigation Speed

Eunhee Lee[0] Dongman Lee Seunghyun Han Soon J. Hyun
School of Engineering, Information & Communications University

요 약

가상 세계의 공유 개념은, 특히 사용자들이 인터넷 같이 대규모 네트웍을 통해 지역적으로 분산된 경우는 복제가 수용할 수 있는 상호작용 성능을 제공하기 때문에 각 사용자의 사이트에 정보를 복제함으로써 확장된다. 그러나, 다수의 동시 갱신은 replicas간의 일관되지 않은 뷰를 일으키게 될 것이다. 따라서, 동시성 제어가 복제자들간에 일관된 상태를 유지하도록 하기 위한 중요한 요소가 된다. 우리는 단지 대상 객체의 주변에 있는 사용자들만이 소유권 요청을 다중 전송하게 하는 확장성 있는 예측기반 동시성 제어 스킴을 제안했었다. 이 작업에서, 우리는 모든 사용자들이 동일한 속도를 가지고 가상 세계를 이동한다고 가정했다. 이것은, 그러나,좀더 사실성을 더하기 위해 사용자가 가상 세계와 상호작용을 할 때 그들의 이동속도를 변경하도록 하는 네트웍 게임같은 네트웍 가상 환경에서는 너무 common하다. 본 논문은 다양한 속도를 가진 사자를 지원하기 위한 확장을 제안한다. 확장된 스킴은 다른 속도의 수만큼의 다중 Entity Radii를 가지며 각 속도를 가진 사용자에게 분리된 큐를 할당한다. 각 큐는 다음 소유자 후보를 예측하기 위해 동시에 예측을 수행하고 선택된 후보들간에서 최소의 Predicted Collision Time을 가지는 최종 후보자가 선택된다. 이는 사용자의 속도에 기반을 둔 적절한 Entity Radius를 사용함으로써 소유권의 timely advanced transfer과, 다른 이동 속도와 latency를 가지는 사용자들 간의 간섭을 줄임으로써 공정(공평)한 소유권 양도, 그리고 불필요한 소유권 전송을 줄임으로써 높은 예측 정확도를 제공한다.

1. 서론

네트웍 가상 환경은 공간, presence, 시간 등의 측면으로 다양한 정도의 공유성을 제공함으로써 분산된 사용자들 간에 실시간 상호작용을 허락하는 소프트웨어 시스템이다 [13]. 환경에서의 공유 개념은 종종 인터넷과 같은 대규모 네트웍을 통해 지역적으로 사용자가 분산되어 있는 경우에 특히, 복제가 수용할 수 있는 상호작용 성능을 제공하기 때문에 각 사용자의 로컬 정보를 복제하도록 함으로써 확장된다. 그러나, 다수의 동시적인 갱신이 발생함으로 인해 복제자들간에 서로 다른 뷰를 일으키게 될 것이다 [1]. 통신 지연이 증가함에 따라, operation들 간의 충돌 가능성 또한 증가하게 된다. 따라서 동시성 제어는 복제자들 간에 일관된 상태를 유지하기 위한 중요한 요소인 것이다. 대부분의 현존하는 분산 가상 환경 시스템들은 간단한 비관적 (pessimistic) 잠금성 동시성 제어를 적용했다. 비관적 스킴은 잠금 허가를 가진 사용자만이 대상 객체와 상호작용을 허용한다 [4, 6, 7, 12, 16]. 이것은 간단하고 강한 일관성을 제공한다. 그러나 이것의 단점은 통신 지연이 높아짐에 따라 네트웍 latency로 인한 소유권 전송 지연을 일으킬 수 있다는 것이다. 낙관적 (optimistic) 동시성 제어와 예측 기반 동시성 제어는 지연을 줄임으로써 최대의 상호작용 성능을 제공하기 위해 제안되었다. 낙관적 동시성 제어는 여러 사용자들이 객체에 대한 일시적인 잠금 허가를 가지도록 하여 동시적으로 객체의 속성을 갱신하도록 한다 [15]. 그리고 나서 committing하기 전에 갱신을 validate하고 만약 충돌이나 잘못된 갱신이 발생했다면 이전 상태로 rollback한다. 예측 기반 동시성 제어는 상용자들에게 실시간 상호작용을 제공하고 소유권 예측을 통해 미리 객체의 소유권을 전송함으로 복구의 필요성을 피한다. 그러나, 기존의 예측 기반 동시성 제어 방식 [9, 10]은 소유권을 원하는 사용자들이 전체 가상 세계에 속한 모든 잠재적인 소유자들에게 소유권 요청을 다중전송하기 때문에 참가자의 수와 객체의 수가 증가함에 따라 확장성을 제공할 수 없게 된다. 이에 우리는 대상 객체 주변에 있는 사용자들에게만 소유권 요청이 다중전송 되도록 하는 확장성 있는 예측 기반 동시성 제어 방식을 제안했었다 [17]. 대상 객체의 주변 영역을 미리 정의하고 이를 Entity Radius라 한다. Entity Radius에 진입한 사용자들만이 묵시적인 소유권 요청으로 객체에 할당된 멀티캐스트 어드레스를 통해 조인 메시지를 다중전송하고 소유권 후보자가 된다. 현재 소유자는 줄어든 메시지수와 함께 예측을 하며, 결국 짧은 요청 처리 시간에 이르게 된다. 소유자는 다음 소유자를 결정하여 제때에 소유권을 전달한다. 이전 작업에서, 우리는 모든 사용자들이 동일한 이동속도로 가상 세계를 이동한다고 가정했다. 그러나, 이는 가상 환경과 상호 작용할 때 더욱 현실감을 더하기 위해 사용자들이 자신들의 이동 속도를 변경시킬 수 있는 네트웍 게임과 같은 네트웍 가상 환경에서는 quite common하다. 본 논문에서는 다양한 이동 속도를 가진 사용자를 지원하기 위한 확장을 제안한다. 이전 작업의 실험을 통해 Entity Radius의 크기가 in time 소유권의 advanced transfer를 위해서는 최대 latency와 최대 이동속도의 곱에 2배가 되어야 한다는 것이 증명되었다 [17]. 만약 다른 속도를 가진 사용자들이 대상 객체에 접근하고 있고 Entity Radius는 최대 속도를 이용해 설정되었다면, Entity Radius의 크기가 커질수록 더 많은 사용자들이 소유자 후보로 받아들여질 것이다. 이것은 예측 수행 시간 - 큐잉 타임 - 을 증가시켜 결국에는 정확한 예측을 방해하여 상호 작용 성능을 감소시키게 된다. 대신, 확장된 방식은 다른 이동 속도의 가지 수 만큼 많은 Entity Radii를 허용하고 각 속도의 사용자들을 위한 분리된 큐가 할당된다. 각 큐는 동시에 다음 소유권 후보자를 예측하고 선택된 후보자들간에 최소의 predicted collision time을 가지는 최종 후보자가 선택된다. 이것은 사용자의 이동 속도에 기반을 한 적절한 Entity Radius를 사용함으로써 timely advanced transfer of ownership과, 다른 속도와 latency를 가진 사용자들 간의 간섭을 줄임으로써 공정한 소유권 양도, 그리고 불필요한 소유권 전송을 줄임으로써 높은 예측 정확성을 제공한다. 성능 평가의 결과들은 확장된 방식이 소유권의 on-time 전송과 가상 세계 내의 객체와 다른 속도를 가진 사용자의 수가 증가하더라도 높은 소유권 예측 확률을 제공함을 보인다. 논문의 나머지는 다음과 같이 구성된다. 2장에서는 기존의 예측 기반 동시성 제어 방식인 PaRADE를 소개한다. 3장은 이전 작업의 핵심 개념들인 Entity-Centric multicast 방식을 설명한다. 4장은 확장된 방식을 자세히 설명하고, 5장에서는 성능 평가의 결과들이 보여진다. 결론은 6장에서 맺는다.

2. 관련연구

DIVE[4], SPLINE[16], CAVERNsoft[7], Virtual Society[6]와 Bricknet[12] 등은 비관적 동시성 제어 방식을 적용했다. 먼저 첫번째 세 시스템들은 lock 전송 관

점에서 마지막 두 시스템이 중앙 제어 방식인 반면 분산된 방식을 사용했다. Lock을 가진 사용자 만이 객체 데이터를 조작할 수 있고 같은 월드의 다른 사용자들에게 그 내용을 전송한다. 객체에 대한 갱신의 충돌이 발생한 경우, 사용자는 lock을 받을 때까지 블락킹(blocking)되는데, 이는 사용자로 하여금 다른 동작들도 금지하게 되므로 문제가 될 수 있다. CIAO[15]는 응답성 문제를 해결하기 위해서 준 낙관적 동시성 제어 방식 (semi-optimistic concurrency control)을 사용했는데, 이 방식에서 사용자는 lock이 없는 상태에서 오직 한 동작을 수행할 수 있다. 후에 만약 lock을 얻는데 실패하게 되면, 객체에 대한 조작을 중지하게 된다. 즉, 이 방식은 상호 작용 성능을 위해 동시적인 갱신으로부터 기인된 일시적인 inconsistent view를 허용한다. 그리고 PaRADE[10]는 예측 기반 동시성 제어 방식을 활용한 첫번째 시스템으로, 소유권 후보자들이 자신들의 객체 조작 시간을 계산하여 미래의 동작을 위해 미리 토큰을 요청하여 실제 갱신 시간 전에 그 후보자에게 토큰이 주어지게 된다. 이 방식은 사용자가 지연없이 객체와 상호작용을 하도록 한다. 그러나, 토큰 요청을 월드에 할당된 multicast address를 통해 전송되기 때문에, 현재 소유자를 포함한 월드 내의 모든 사용자들이 모든 다른 사용자들의 소유권 요청을 수신하게 된다. 따라서, 각 소유자들은 자신과 관련없는 요청에 대해서도 소유권 검사 및 처리 과정을 거쳐야 할 것이다. 이는 분산 가상 환경의 크기가 커짐에 따라서, 잠재적인 소유권 후보자들이 커지게 됨으로 incoming 메시지 수 역시 증가하게 되고 네트워크도 과부하될 것이다. 결국, 이는 시기 적절한 소유권 전송에 방해가 되어 잘못된 예측을 야기시키게 된다.

3. 객체 중심 예측 기반 동시성 제어 (ENTITY-CENTRIC PREDICTIVE CONCURRNECY CONTROL)

성공적인 소유자 예측의 핵심은 다음 소유권 후보자의 정확한 예측과 객체에 도달하기 직전에 소유권을 사용자에게 미리 전송하는 것, 그리고 잘못된 예측이 발견되었을 때 즉각적으로 바로 잡는 것이다. 따라서, 예측 스킴은 2가지 구성요소인, 사용자 예측과 복구로 구성된다. 정확한 예측을 위해서 가상 월드내의 사용자들 중에서 잠재적인 소유권 후보자들이 누구인지를 가려내는 것과 그 후보자들 중 다음 소유자를 결정하고 소유권을 전송하는 시간을 결정하는 것이 매우 중요하다. 복구는 잘못된 예측으로부터 빠르게 회복시키는 방법을 다루게 된다.

3.1 Entity Radius

예측 기반 동시성 제어에서, 소유권에 대한 요청을 미리 하는 것은 현재 소유자가 다음 소유자를 예측하고 소유권을 전송하는데 충분한 시간을 가질 수 있도록 하기 위한 것이다. 이를 위해서, 우리는 객체와 관심을 가진 사용자들 간의 지역적 근접성을 이용하여 대상 객체 주변에 특정한 영역을 정의하여, 이를 Entity Radius라 불른다 [17]. Entity Radius는 공간적 관심 영역을 나타내는 것으로, 객체의 Entity Radius에 진입한 사용자는 객체에 대한 조작 의도가 있는 것으로 간주되고, Entity Radius에 할당된 multicast address로 소유권 요청 메시지를 multicast하게 된다. Entity Radius의 크기는 우리의 이전 연구에서 행해진 실험을 통한 결과를 기반으로 해서 다음과 같이 설정된다.

$$Size\ of\ Entity\ Radius = 2 * maximum\_latency * navigation\_speed$$

이 식에서, 두번째와 세번째 항목은 대상 객체에 좀더 멀리 떨어져 있는 소유권 후보자들이 현재 소유자에게 소유권 요청을 할 수 있도록 충분한 시간을 제공하기 위한 것이다. 그리고, 세번째 항목은 공정하고 정확한 예측과 다음 소유자에게 소유권을 양도하기 위한 충분한 시간을 위해 도입된 것이다.

3.2 소유권 예측과 복구

현재 소유자가 대상 객체에 대한 소유권을 릴리즈하기 전에, 다음 소유자에 대한 예측을 해야만 한다. 사용자는 대상 객체의 Entity Radius에 진입하자 마자, 묵시적인 소유권 요청으로 대상 객체 multicast address에 조인 메시지를 보낸다. 그러면, 현재 소유자는 새로운 조인 메시지에 대한 사용자 정보를 소유권 후보자 큐에 추가시키고, 그가 소유권을 릴리즈하기 전에 다음 소유자를 예측할 때 대상이 된다. 이 소유권 후보자 큐에 유지되는 정보는 후보자 ID, 현재와 이전의 위치값, 이동 속도, 객체와의 거리, 이동 방향, 그리고 객체와의 예측된 충돌 시간(Predicted Collision Time)이다. 후보자들이 움직일 때마다, 이 큐의 정보 또한 갱신되게 된다. 가상 월드에서, 사용자의 위치는 같은 월드내의 다른 사용자들에게 서로를 보이도록 multicast하는 것이다. 예측된 충돌 시간은 객체와의 현재 거리를 이동 속도로 나눔으로 해서 계산되고, 이동 방향은 사용자의 현재 위치가 객체와의 이전 위치보다 가까우면 POSITIVE값을 갖게 되고, 그렇지 않으면 NEGATIVE를 갖게 된다. 현재 소유자는 소유권을 후보자들 중에서 POSTIVE 이동 방향을 가지면서, 예측된 충돌시간이 가장 가까운 후보자를 다음 후보로 결정한다. 잘못된 예측은 반복된 소유권 전송을 일으켜 응답성을 감소시키므로 최소화 시켜야 하기 때문이다. 이런 잘못된 예측은, 예측된 소유자가 소유권을 받고도 객체와 상호작용을 하지 않고 지나치거나, 예측된 소유자가 객체에 도달하기 전에 다른 사용자가 객체와 충돌을 일으킬 경우 발생하게 된다. 이 때 잘못된 예측 확률을 줄이기 위해서, 예측된 충돌시간에서 전송 지연시간을 뺀 시간에 소유권을 전송하므로 잘못된 예측에 대한 복구를 위한 시간을 기다리게 되는 것이다.

4. 다른 이동 속도를 가진 사용자를 지원하기 위한 확장

이 장에서는 이전 장에서 설명된 객체 중심 스킴의 확장 알고리즘과 디자인 고려사항을 설명한다. 확장은 다른 이동 속도의 사용자의 지원이다. 예를 들어, 네트웍 게임이나 가상 커뮤니티 같은 가상 현실 애플리케이션들은 사용자가 다양한 이동 속도들 중에서 하나를 동적 내지는 정적으로 선택하도록 한다. 만약 각 사용자들이 네트웍 지연, 이동 속도, 그리고 객체에 대한 관심 정도 면에서 서로 다른 정도를 가진다면, Entity Radius는 이런 다른 특징들을 모두 반영해야만 한다. 이전 연구에서의 Entity Radius의 크기는 적절한 소유권 전송과 올바른 예측을 위한 충분한 여유를 얻기 위해 최대 지연 시간과 최대 이동 속도를 반영했다 [4]. 만약 서로 다른 이동 속도를 가진 사용자들이 객체에 접근하고 있고 Entity Radius는 최대 속도로 설정되었다고 하면, 크기가 너무 커지게 되기 때문에 보다 많은 사용자들이 후보자로 받아들여지게 될 것이다. 특히, 느린 속도로 이동하는 사용자들은 너무 이른 소유권 요청이 보내지게 될 것이고, 결국에는 큐잉 사이즈를 증가시키게 된다. 즉, 이는 현재 소유자의 큐 처리 시간을 증가시키고, 예측을 지연시키며, 최종적으로는 상호 작용 성능을 감소시키게 될 것이다. 이런 다양한 이동 속도를 가진 사용자를 지원하기 위해서 우리는 다른 속도의 수 만큼의 다중 Entity Radii를 미리 정의했다. 하나의 Entity Radius는 각 이동 속도를 가진 사용자들을 위해 정의되고, 현재 소유자는 각 Entity Radius를 위한 분리된 후보자 큐를 관리하게 된다. 이것은 예측에 대한 처리 시간에 영향을 미치는 큐잉 타임을 줄이게 된다. 다음 후보자 예측시에, 각 큐는 동시에(*in parallel*) 큐 마다의 잠재적인 다음 소유자를 예측하게 되고 이들이 실제 후보자가 되어 최종 하나의 다음 소유자가 선택되게 된다. 이로써 다음 소유자의 소유권의 시기 적절한 전송을 하게 된다. 예를 들어, 이동 속도가 1에서 4까지 4종류가 지원 가능하고, 최대 지연 시간이 100인 가상 현실 애플리케이션을 가정하자. 그림 1은 이 가정에 대한 다중의 Entity Radii 구조를 보여준다. 원, 삼각형, 사각형, 사다리꼴은 서로 다른 속도를 가진 사용자를 나타내고, 원기둥은 대상 객체, 그리고 육각형은 현재 소유자를 나타낸다. 이동 속도가 1-4까지 4가지 이므로, 그에 해당하는 Entity Radius도 실선, 점선등, 서로 다른 크기의 4개가 존재하며, 각 사용자는 자신의 이동 속도에 맞는 Entity Radius를 이용한 동시성 제어에 참여하여 된다. 그림에서, "A"로 표시된 사용자들은 자신과 맞지 않은 Entity Radius에 충돌을 일으켰으므로, 소유권 요청을 보내지 않는다. 그리고 "B"로 표시된 사용자들만이 자신의 이동 속도에 맞는 Entity Radius에 진입하게 되므로, 해당 객체에 대한 소유권 요청을 객체에 할당된 multicast 그룹에 조인하므로써 보내게 된다. 그리고 "C"로 표시된 사용자는 자신의 이동 속도에 맞는 Entity Radius내에 진입해 있으므로, 이미 해당 객체에 대한 후보자가 되어 있다. 현재 소유자는 이런 4가지 이동 속도를 가진 각 후보자들을 각각 관리하기 위한 별도의 4개의 후보자 큐를 유지하고, 조인 메시지가 오면 그에 맞는 큐에 후보자를 등록하고, 위치가 이동할 때마다 정보를 갱신하며, 소유권을 릴리즈 시킬 때 이 4개의 후보자 큐를 이용해서 최종의 다음 소유자를 예측하게 되는 것이다.

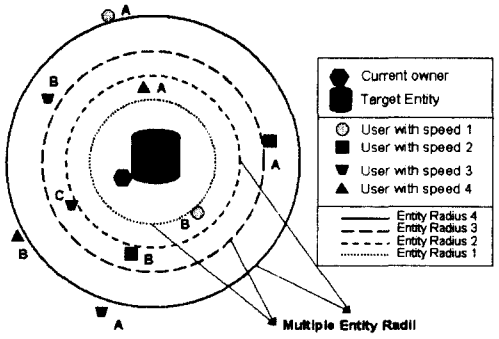

그림 1. Structure of the enhanced scheme with Multiple Entity Radius

5. 실험 결과

우리는 시뮬레이션을 통해서, 이전 연구와 비교한 본 확장 연구의 일반성과 확장성을 분석했다. 본 시뮬레이션의 확장된 스킴에서는 이동 속도는 4가지 - 1, 2, 3, 4 - 가 있고, 최대 지연 시간은 100으로 가정하여, 4개의 Entity Radius가 존재하며, 각각의 크기는 200, 400, 600, 800이 된다. 그리고 이전 연구에서는 최대 속도에 의해 설정된 크기가 800인 1개의 Entity Radius가 존재한다. 컴퓨터에 의해 생성된 아바타들과 객체들이 널리 분산되어 있는 상태의 간단한 분산 가상 환경을 시뮬레이션 했다. 확장된 스킴의 일반성을 증명하기 위해서, 우리는 다양한 속도를 가진 사용자들의 분포측면에서 2가지의 경우 - 모든 속도를 가진 사용자의 분포 수가 동일한 경우와 특정한 속도를 가진 사용자의 수가 다른 속도의 분포수보다 5배 많은 경우 - 로 두 스킴을 실험했다. 그리고 확장성을 사용자의 수가 증가함에 따른 두 스킴의 정확한 예측도를 측정했다.

5.1 일반성 (Generality)

정확한 예측은 소유권이 대상 객체에 맨 먼저 도달하는 사용자에게 정확하게 배달되었음을 의미한다. 그림 2는 이에 대한 결과를 보여준다.

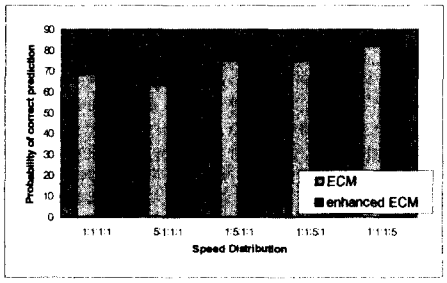

그림 2. 이동 속도의 분포에 따른 평균 예측 정확도

그래프에서 느린 속도를 가진 사용자가 많음에 따라 이전 스킴의 정확한 예측도가 감소됨을 볼 수 있다. 이는 Entity Radius의 크기가 증가함에 따라 복구의 확률이 증가하기 때문이다. 다시 말해, 대상 객체에 가까이 있지도 않은 사용자들을 포함해 더 많은 사용자가 소유권 후보자가 된다는 것이다. 잘못된 다음 소유자 예측으로부터의 복구 또한 더 빈번하게 발생한다. 이것은 결국 상호 작용 성능을 감소시키게 되는 불필요한 소유권 전달을 증가시킨다. 그러나, 확장된 방식은 더 높고 정확한 소유자 예측률을 제공하고 이동 속도에 따른 다양한 사용자의 분포에 관계없이 적절한 소유권 전달 (on-time delivery)을 지원한다. 이는 주어진 속도를 가진 사용자들이 그에 맞는 Entity Radius에 대한 분리된 큐에서 처리되기 때문이고, 또한 소유자 예측도 시간내에 행해질 수 있으므로 잘못된 예측의 기회가 줄어들기 때문이다.

5.2 확장성

확장성있는 동시성 제어를 위해서는 사용자의 수가 증가하더라도 정확한 예측율이 굉장하게 감소되지 말아야 한다. 우리는 사용자 수를 100명에서 400명까지 증가시키면서 이전 연구 방식과 확장된 방식에서의 예측 정확도를 측정했다. 그림 3은 결과를 나타낸다.

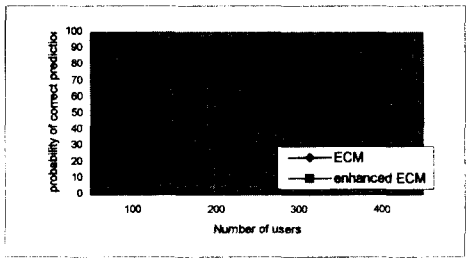

그림 3. 사용자수의 증가에 따른 예측 정확도

서로 다른 이동 속도를 가진 사용자의 수가 증가함에 따라, 이전 연구 방식에서는 예측 정확도가 약간의 감소를 나타낸다. 이것은 예측에 영향을 미치는 큐잉 타임이 큐의 크기에 의존하고, 이 큐의 크기는 참여자들의 수에 의존하기 때문이다. 증가된 큐잉 타임은 예측 처리 시간의 증가를 야기시키게 된다.

반면, 본 확장된 방식은 참여자들의 수에 관계없이 일정한 예측 정확도를 지원한다. 이것은 각 참여자들이 자신들에 맞는 시점에 소유권 요청을 보낼 수 있고, 또한 소유권 요청들이 각 이동 속도에 맞는 분리된 큐에서 큐잉되고 예측을 위해 처리되기 때문이다.

6. 결론

상호 작용 성능 (Interaction performance)는 동시성 제어에 의존하게 된다. 예측 기반 동시성 제어 방식의 효율성은 [10]에서 증명되었다. 적절히 이른 소유권 요청과 소유권 전송은 예측 기반 동시성 제어에 영향을 미치게 된다. 우리는 이전 연구에서 네트웍 지연과 최대 이동 속도를 고려한 Entity Radius를 제안했다 [17]. 그러나, 사용자들이 서로 다른 이동 속도를 가지게 되고, Entity Radius가 최대 속도에 의해서 설정이 된다면, Entity Radius의 크기가 커짐에 따라 관계없는 사용자들을 포함하여 더 많은 사용자들이 소유권 후보자가 될 것이다. 이것은 소유권 예측을 위한 처리 시간의 증가를 야기시켜 결국에는 상호 작용 성능을 감소시키게 된다. 본 연구의 확장된 방식은 예측 정확도에 영향을 미치는 Entity Radius를 최대 속도로 설정된 한 개만 두는 것이 아니라, 허용되는 이동 속도의 수만큼 다중의 Entity Radii를 두고, 또한 그에 맞는 후보자 큐도 분리하여 여러 개 유지하도록 한다. 이것은 서로 다른 통신 지연 및 이동 속도에 관계없이 공정한 소유권 양도를 이루게 된다. 각 Entity Radius는 대규모 분산 가상 환경에서의 서로 다른 이동 속도를 가진 사용자들의 적절한 소유권 요청의 책임을 지게 되고, 이동 속도에 따라 분리된 각 큐는 예측시 큐잉 타임을 줄이는게 일조를 하게 된다.

7. 참고 문헌

[1] S. Bhola, S. Banavar, and M. Ahmad, "Responsiveness and Consistency Tradeoffs in Interactive Groupware," ACM CSCW'98, Washington, November 1998.

[2] S. Greenberg and D. Marwood, "Real Time Groupware as a Distributed System: Concurrency Control and its Effect on the Interface," ACM CSCW'94, North Carolina, October 1994, pp. 207-217.

[3] C. Greenhalgh, J. Purbrick, and D. Snowdon, "Inside MASSIVE-3: Flexible Support for Data Consistency and World Structuring," CVE 2000, ACM

[4] O. Hagsand, "Interactive Multiuser VEs in the DIVE System," IEEE *multimedia*, 1999, pp. 30-39.

[5] O. Hagsand, R. Lea, and M. Stenius, "Using Spatial Techniques to Decrease Message Passing in a Distributed VE System," *Proceedings of the second symposium on Virtual reality modeling language*, 1997, pp. 7.

[6] R. Lea, Y. Honda, K. Matsuda, O. Hagsand, and M. Stenius, "Issues in the design in a scalable shared virtual environment for the Internet," HICSS, 1997.

[7] J. Leigh, A. Johnson, T. DeFanti, "CAVERN: A Distributed Architecture for Supporting Scalable Persistence and Interoperability in Collaborative Virtual Environments," Journal of Virtual Reality Research, Development and Applications, the Virtual Reality Society, Vol. 2.2, 1997, pp. 217-237.

[8] M. Macedonia, and M. Zyda, "Taxonomy for Networked Virtual Environments," IEEE multimedia, 1997, pp. 48-56.

[9] D. Roberts, A. Richardson, P. Sharkey, and T. Lake, "Optimising Exchange of Attributed Ownership in the DMSO RTI," SISO'98

[10] D. Roberts, P. Sharkey, and P. Sandoz, "A Real-time, Predictive Architecture for Distributed Virtual Reality," *Proc. 1st ACM Siggraph Workshop Simulation & Interaction in Virtual Environments*, Des Monies, Iowa, pp. 279-288, July 1995.

[11] P. Sandoz, P. Sharkey, and D. Roberts, "Collision prediction of a moving object within a virtual world," VR World '96, February 13-15th, Stuttgart, Germany

[12] G. Singh, L. Serra, W. Png, and H. Ng, "BrickNet: A Software Toolkit for Network-Based Virtual Worlds," Presence, MIT Press, Vol. 3, No. 1, 1994, pp. 19-34.

[13] S. Singhal and M. Zyda, "Network Virtual Environments: Design and Implementation," Addison Wesley, ACM Press, July 1999.

[14] M. Stytz, "Distributed Virtual Environments," IEEE Computer Graphics and Applications, May 1996, pp. 19-31.

[15] U. Sung, J. Yang, and K. Wohn, "Concurrency Control in CIAO," IEEE VR'99, pp. 22-28

[16] R. Waters, D. Anderson, and D. Schwenke, "Design of the Interactive sharing Transfer Protocol," IEEE WETICE, 1997, pp.140-147.

[17] J. Yang, and D. Lee, "Scalable Prediction Based Concurrency Control for Distributed Virtual Environments," IEEE VR'00, March 2000.