클라이언트-서버기반 분산가상환경에서의 지연예측을 통한 효율적 공유상태관리

심 광현, 최 병태, 김 종성, 오 원근
한국전자통신연구원 가상현실연구센터
전화 : 042-860-6673 / 핸드폰 : 011-9804-3687

An Effective Shared-State Management using Network Delay Estimation in Client-Server-Based Networked Virtual Environment

Kwang-Hyun Shim, Byung-Tae Choi, Jong-Sung Kim, Weon-Geun Oh
VR R&D Center, CSTL, ETRI
E-mail : shimkh@etri.re.kr

Abstract

This paper presents a new DR(Dead Reckoning) algorithm in client-server-based networked virtual environment using network delay estimation. In the algorithm, a new update packet is sent to server (or client) whenever the difference of current real value and tracking value after network delay is larger than threshold. To confirm the proposed algorithm, a test network game was implemented. Through iterative field tests, we knew that this algorithm provides fair service and stability.

I. 서론

분산가상환경이란 네트워크를 통해 다수의 사용자가 협력하여 공동으로 작업(설계, 게임 등등)을 할 수 있는 환경을 말한다. 분산가상환경에서는 다른 사용자가 한 사용자가 작업한 결과가 다른 사용자들에게 반영되도록 하기 위해 공유하고 있는 객체를 조작하여 상태가 변화되면 그 객체를 공유하고 있는 모든 다른 사용자에게 변화된 상태를 보내주게 된다. 그러나, 네트워크의 지연으로 인해 다른 사용자들은 지연시간 후에 변화를 인지하게 되므로 사용자들 간에 공유상태가 불일치해질 뿐 만 아니라 네트워크의 한정된 자원을 고려하여 변화된 상태는 샘플링되어 보내지게 되므로 변화된 상태를 받는 사용자들은 연속적이 아닌 샘플링된 결과만을 보게 된다. 이것은 분산가상환경이 직면하고 있는 문제점들 중의 하나로써 지금까지 이를 해결하기 위한 많은 연구가 진행되어 왔는데 그 대표적인 것이 DR(Dead Reckoning) 알고리즘이다.

DR 알고리즘은 크게 갱신패킷(update packet)을 받는 쪽에서는 샘플링되어 과거에 받은 갱신패킷내의 정보(예: 위치, 속도, 가속도등)를 사용하여 현재시각의 상태를 예측하는 트래킹 알고리즘과 또 다시 새로운 데이터를 받으면 그것을 고려하여 새롭게 예측된 상태로 연속적으로 수렴하기 위한 수렴 알고리즘으로 구성된다. 이를 사용하여 사용자들은 오차가 있기는 하지만 보다 실시간에 그리고 자연스럽게 상태변화를 인지하게 된다. 초기의 Amaze와 같은 시스템에서는 상태예측을 위한 트래킹 알고리즘으로 1차 함수를 이용하였다.

$$x(t) = x(t_0) + x'(t_0)t$$

여기서, t_0는 가장 최근에 갱신패킷을 받은 시간이고 $x(t_0)$와 $x'(t_0)$는 그때 받은 정보들이다. 이후, SIMNET(U.S. Army's Simulation Networking System) [3]이나 DIS [4]등과 같은 상용시스템에서는 트래킹 알고리즘으로 2차 함수를 사용하였다.

$$x(t)=x(t_0)+x'(t_0)t+\frac{1}{2}x''(t_0)t^2$$

여기서는 위에 비해 $x''(t_0)$값을 추가로 더 받게 된다. 그리고, 근래에는 탱크, 비행체 등등 각각의 공유 객체들의 특성에 맞게 특화된 DR 알고리즘들이 많이 연구되고 있다.

이와 달리, PARADISE 에서는 속도와 가속도를 제외한 상태값 만을 받아서 그것의 이력으로부터 속도와 가속도를 계산하여 트래킹 알고리즘을 만들고 수렴 알고리즘으로는 상태에서 따라 1차 함수와 2차 함수를 스위칭하여 사용하는 방법을 제안하였다.

한편, 보내는 쪽에서는 문턱값(Threshold)을 두어 현재의 상태값와 가장 최근에 다른 사용자에게 보낸 상태로부터 예측된 상태값과의 차이가 문턱값 이상이 되었을 때에만 다른 사용자에게 상태변화를 보내줌으로써 어느 정도 전송율을 제어해 주게 된다.

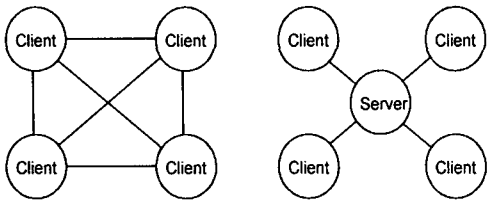

그림 1 Peer-To-Peer(왼쪽)와 Client-Server

그러나, 이러한 일련의 연구결과들은 Peer-to-Peer 형태의 분산가상환경을 가정하여 만들어진 것일 뿐 만 아니라 그 결과를 분석해 보면 상태변화가 다른 사용자에게 전송되는 데까지의 네트워크지연으로 인해 상태변화를 받은 쪽에서는 받을 당시의 실제값과 예측값의 차이는 문턱값보다 더 커지게 되는데 이것은 네트워크지연이 큰 사용자일수록 더 커지게 되어 문턱값을 제대로 보장을 못해주게 된다. 즉, 네트워크지연이 큰 사용자일수록 공유상태를 더욱 부정확하게 인지하고 있게 되어 저질의 서비스를 받게되는 효과를 보이게 된다.

이에, 본 연구에서는 클라이언트-서버 분산가상환경에 적합하고 모든 사용자에게 공평한 서비스 및 오차의 문턱값을 보장해 줄 수 있는 네트워크지연을 고려한 효율적인 DR 알고리즘을 제안한다.

II. 시스템구성

2.1 전체 시스템 구성

i) 전체 분산가상환경 시스템은 하나의 서버에 N개의 클라이언트가 연결되어 있다고 가정한다.

ii) 클라이언트와 서버사이에 교환되는 사건들을 다음과 같은 두 가지로 분류를 한다. 첫째는, 하나의 클라이언트에서 발생되어 서버로 전송되고 다시 다른 클라이언트에게 전송되는 사건들이다. 이것의 예로는 클라이언트가 어떤 특정 객체를 조작하거나 클라이언트가 가상공간 내에서 이동을 하거나 하는 등의 사건들이 이것에 해당된다. 두 번째로, 서버에서 클라이언트로만 전송되는 사건들이다. 이것의 예로는, 가상환경 내에서 자기 자신만의 특정한 다이나믹스를 가지면서 움직이는 객체들에서 자동으로 생성되는 사건들이 해당된다. 특히, 본 논문에서는 서버와 클라이언트들간에 교환되는 여러 가지 종류의 사건들 중에서 특히 객체들의 이동에 따른 사건만을 다룬다.

iii) n번째 클라이언트에서 서버까지의 네트워크지연을 τ_n^{cs} 라고 정의하고 서버에서 n번째 클라이언트까지의 네트워크지연을 τ_n^{sc} 이라고 정의한다. 여기서, $n=1,2,\cdots,N$

iv) 모든 서버와 클라이언트들의 시스템 시간은 NTP (Network Time Protocol)에 의해 모두 동기화되어 있다고 가정한다. 즉, 이것은 모든 시스템의 시간이 일치한다는 것을 의미한다.

v) 트래킹 알고리즘과 수렴 알고리즘은 모든 조합이 사용 가능하지만 일단 1차 함수를 사용한다고 가정한다. 이때, 클라이언트들과 서버사이에서 교환되는 갱신패킷들은 다음과 같은 필드들로 이루어진다.

(이름, 시간, ID, 위치, 속도, 네트워크지연)

여기서, 이름은 미리 정의된 사건의 종류를 의미하고, 시간은 그 사건의 갱신시간을 의미하고, ID는 사건이 일어난 대상 객체의 공유 ID를 의미하고, 위치는 객체의 현 위치를 (x,y,z)좌표로 표시한 것이고, 속도는 객체의 이동속도를 3차원 벡터로 표시해준 것이고, 마지막으로 네트워크지연은 클라이언트에서 서버로 보내지는 갱신패킷에서는 서버에서 클라이언트까지의 평균 네트워크지연을 클라이언트가 계산하여 보내주고, 서버가 클라이언트에게 보내는 갱신패킷에서는 그 클라이언트에서 서버를 거쳐서 다른 클라이언트로의 최대 네트워크지연을 서버가 계산하여 보내준다.

vi) 클라이언트와 서버사이의 네트워크지연의 갱신패킷의 도착시간에서 갱신패킷의 두 번째 필드인 사건 갱신시간을 빼 주면 된다. 이 때, 순간적으로 랜덤하게 변화되는 실제 네트워크의 상황을 고려하여 새로운 네트워크지연의 계산은 과거의 값들과 평균을 취해서 얻는다.

$$\tau_{new}=(1-\alpha)*\tau_{arrived}+\alpha*\tau_{old}, \quad (2\text{-}1)$$

여기서, $0\le\alpha<1$이다. α가 1에 가까워질수록 과거의

네트워크지연의 영향이 커지게 되고 순간적인 변화에 덜 민감해 지는 등 고역(High-Pass) 필터 효과를 나타내는 반면, 0에 가까워질수록 새로 추가되는 네트워크 지연의 영향이 커지게 되어 변화에 대해 빠른 응답을 보이게 된다. 여기서는, 네트워크의 상황의 변화속도는 현재 시스템의 최소 상태변화속도에 비해 충분히 느리다고 가정한다.

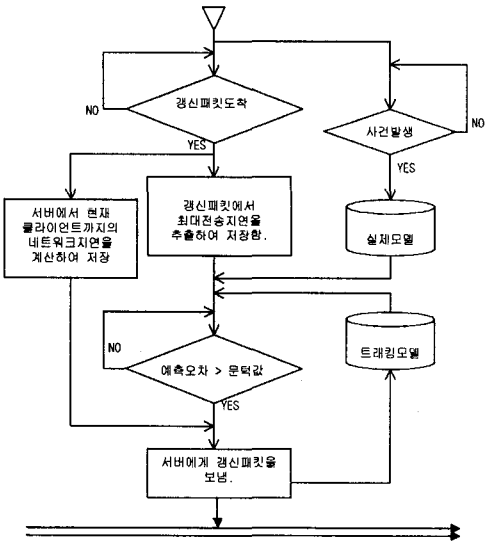

그림 2 클라이언트에서의 DR 메카니즘

2.2 클라이언트에서의 DR 메카니즘

i) n번째 클라이언트가 서버로부터 갱신패킷을 받으면 갱신패킷의 도착시간과 갱신패킷내의 사건 갱신시간 및 식 (2-1)을 이용하여 서버로부터 n번째 클라이언트까지의 전송지연 τ_n^{sc} 을 계산하여 서버에게 다음 갱신패킷을 보낼 때 같이 보낸다. 여기서, $n=1,2,\cdots,N$

ii) 갱신패킷의 마지막 필드인 n번째 클라이언트에서 서버를 거쳐 다른 클라이언트까지의 최대 전송지연 $\tau_n^{\max}$을 추출하여 저장한다.

$$\tau_n^{\max} = \tau_n^{cs} + \max\{\tau_m^{sc}, \text{all } m \neq n\}$$

iii) n번째 클라이언트에서 발생되는 사건에 대해서 현재시간 t 에서 실제모델에서의 실제치 $P(t)$와 최대전송지연을 갖는 클라이언트의 트래킹모델에서의 예측치 $P_e(t+\tau_n^{\max})$와의 차이가 문턱값 P_{th} 이상이 되면 서버에게 갱신패킷을 보내서 클라이언트들이 다시 트래킹모델을 수정하게 한다. 여기서, $n=1,2,\cdots,N$

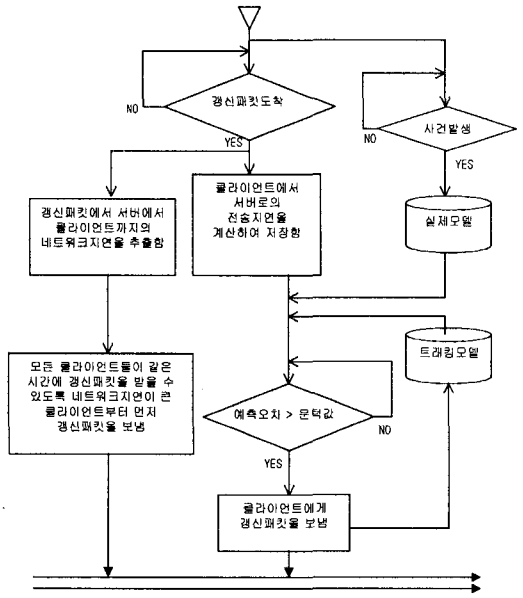

그림 3 서버에서의 DR 메카니즘

2.3 서버에서의 DR 메카니즘

i) 서버는 클라이언트들로부터 갱신패킷이 도착하기를 기다리다가 n번째 클라이언트로부터 갱신패킷이 도착하면 먼저 갱신패킷이 도착한 시간에 갱신패킷내에 있는 발생시간을 빼서 n번째 클라이언트에서부터 서버까지의 네트워크전송지연 τ_n^{cs} 을 계산하여 저장한다. 이것 또한 식 (2-1)을 사용한다.

ii) 갱신패킷의 마지막 필드인 서버에서 n번째 클라이언트까지의 전송지연 τ_n^{sc} 을 추출하여 저장한다.

iii) n번째 클라이언트로부터 받은 갱신패킷은 서버에서부터 전송지연이 가장 큰 클라이언트를 기준으로 해서 받은 것이기 때문에 최대지연의 클라이언트가 갱신패킷을 받게 되는 시간과 동일한 시간에 다른 클라이언트들도 받도록 보내는 시간을 네트워크지연에 따라 조절한다. 즉, 클라이언트들이 받는 시간을 모두 같게 함으로써 클라이언트들이 공평한 서비스를 받도록 할 수가 있다. 즉, n번째 클라이언트에게서 받은 갱신패킷을 m번째 클라이언트에게는 $\tau_n^{\max} - \tau_n^{cs} - \tau_m^{sc}$ 시간 후에 보내게 된다($m \neq n$). 이 때 만약 갱신패킷을 보내기 전에 새로운 갱신패킷이 또 들어오면 기존의 갱신패킷을 새로 들어온 갱신패킷으로 대치하여 같은 시간에 보내게 된다. 이것은 서버가 클라이언트에게 보내는 사건들을 필터링하는 효과를 가져온다.

iv) 서버에서 발생되는 사건에 대해서 실제모델에서의 실제치와 예측을 하는 클라이언트까지의 전송지연을 고려한 트래킹모델에서의 예측치와의 차이가 문턱값 이상이 되면 그 클라이언트에게 갱신패킷을 보내고 트래킹모델을 수정한다.

III. 실험

본 연구에서 제안된 Dead Reckoning 알고리즘을 실험하기 위해서 다음과 같은 간단한 자바기반 네트워크 게임 프로그램을 구현하였다.

i) 분산가상환경에서 서버와 클라이언트들 간의 다양한 방식으로의 데이터 교환, 이를 이용한 세션관리, 사용자관리, 공유객체관리, 비동기사건처리 등등의 기능을 제공하는 분산가상환경을 위한 네트워크 프레임워크로는 현재 과제로써 개발 중에 있는 SHINE(SHared INternet Environment)을 사용하였다.

ii) SHINE을 기반으로 하여 본 연구에서 제안한 알고리즘 및 기타 여러 분산가상환경의 테스트를 위해서

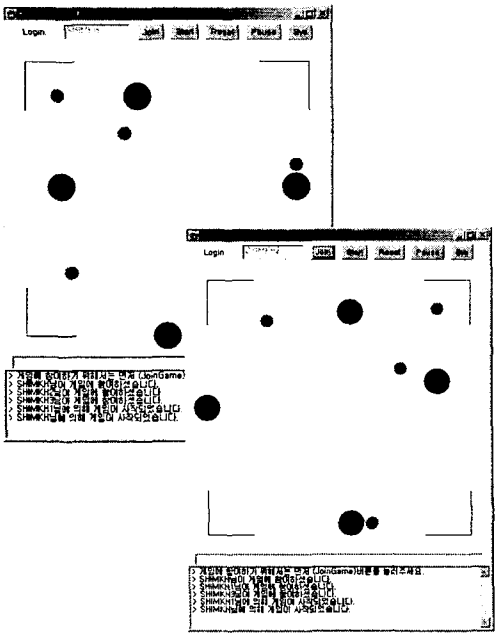

그림 4 네트워크 데모게임의 실행화면

아래와 간단한 네트워크 게임을 개발하였다. 게임의 규칙을 간단히 살펴보면

i) 각각의 사용자는 마우스를 이용하여 자신의 라켓(화면의 아래쪽에 있는 큰 원)을 움직여서 공(화면에서 작은 원)들이 자기 구역이 들어가지 않도록 막는다.

ii) 공이 상대방의 영역에 들어가면 자신의 점수가 +1이 되고 자기 구역으로 들어가면 -1이 되고 먼저 10점을 얻는 사용자가 게임을 승리하게 된다.

현재 XML문서로 되어 있는 게임구성에 따라 정의되어 있는 사용자만큼 한 게임에 참여가 가능하고 그 이상으로 참여하는 사용자들을 위해서는 또 다른 게임 세션이 생성되게 된다. 즉, 다중세션을 지원한다.

현재 이 프로그램의 클라이언트 부분은 자바 애플릿으로 구현되어 있어서 조만간 웹브라우져를 통해 테스트가 가능하도록 할 예정이다.

IV. 결론

본 논문에서는 네트워크전송지연 예측을 통한 Dead Reckoning 알고리즘을 제안하였고 이를 네트워크 게임에 적용시켜보았다. 본 알고리즘에서는 네트워크지연을 고려하여 계산한 예측치을 가지고 오차를 구하므로 오차의 문턱값이 통계적으로 보장된다. 특히, 소그룹의 반복적인 필드 테스트를 통해 제안된 DR 알고리즘이 없는 상황에서는 네트워크지원이 작은 사용자일수록 평균적으로 상대적으로 유리하여 높은 점수를 얻는다는 것을 볼 수 있었고 제안된 알고리즘을 적용시켰을 때에는 평균적으로 모두 비슷한 점수를 얻음으로써 사용자들에게 보다 공평한 서비스를 제공해 줌을 알 수 있었다.

참고문헌

[1] Singhal S. and Michael. Zyda, "Networked Virtual Environments: Design and Implementation," pp.101-146, ACM Press.

[2] Cooke, Joseph M., et al. "NPSNET: Fight Simulation Dynamic Modeling Using Quaternions," Presence: Teleoprators and Virtual Enviroments, pp 404-420, Fall 1992.

[3] Pope, Arthur, "The SIMNET Network and Protocols," Tech. Report 7102, BBN Systems and Technologies, July 1989.

[4] "IEEE Standard for Distributed Interactive Simulation-Application Protocols," IEEE Std 1278.1-2, Sep. 1995.

[5] Singhal, S, "Effective remote modeling in large-scale distributed simulation and visualization environments," Ph.D dissertation Dept. of Computer Science, Stanford University Aug. 1996.