분산 가상현실을 위한 계층적 QoS 지원 기법
(Hierarchical QoS Architecture for Virtual Dancing Environment)

김 진 용 † 원 유 집 †† 김 범 은 † 박 종 일 ††† 박 용 진 ††
(Jin Ryong Kim) (Youjip Won) (Beomeun Kim) (Jongil Park) (Yongjin Park)

요 약 시스템과 네트워크 클라이언트의 제한된 리소스를 이용하여 다수의 사용자가 참여 가능하고 사용자의 QoS를 만족시키는 분산 가상 시스템에 대한 연구가 활발히 진행되고 있다. 이러한 분산 가상 시스템을 구현하기 위해서는 사용자의 증가에 따른 네트워크 트래픽 증가 문제, 네트워크 지연으로 발생하는 클라이언트간의 동기화 문제, 가상 환경 렌더링에 따른 시스템 부하 등의 문제를 해결해야 한다. 본 논문에서는 3인칭 시점의 가상 댄싱 공간을 구성하여 아바타간의 상호 작용을 지원하고 사용자가 느끼는 화질에 따라 네트워크를 통해 전송되는 데이타의 양을 달리하는 계층적 QoS 모델을 제안한다. 계층적 QoS 모델은 화면을 보는 사용자의 시점에 기반하여 가상 댄싱 공간에서 보여지는 아바타의 화질을 조절하는 Visual QoS와 아바타의 영향 범위 및 아바타간의 거리에 의해 사용자에게 들리는 소리의 크기를 조절하는 Audio QoS로 구성된다. 이 기법의 효율성을 검증하고자 Sun사의 JMF와 Java3D를 이용하여 사용자의 실영상과 3D CG 아바타가 참여하는 가상 댄싱 공간을 구현한다. 본 논문에서 제안한 계층적 QoS 모델이 다수의 아바타가 참여하고 아바타의 밀집도가 높은 가상 댄싱 공간을 효과적으로 지원할 수 있다는 것을 TIE 시스템의 성능 실험과 시뮬레이션을 통해 확인한다.

키워드 : 서비스 품질, 가상현실, 멀티미디어 스트리밍, 네트워크, 멀티캐스트, 아바타

Abstract In this paper, we present the virtual dancing studio for distributed virtual environment. In this system, geographically distributed user shares the virtual dancing hall and interacts with each other. The participating object can be a graphical avatar or a live video stream. It allows the coexistence of graphic objects and real images in the shared virtual space. One of the main technical challenges in developing the distributed virtual environment is to handle excessive network traffic. In an effort to effectively reduce the network traffic, we propose a scheme to adjust the QoS of each object with respect to the distance from the observer in the virtual space. The server maintains the QoS vector for each client's shared space and controls the packet traffic to individual clients based on its QoS vectors. We develop a proto-type virtual dancing environment. Java based development enables the client to be platform independent. The result of experiment shows that the adoption of hierarchical QoS management significantly reduces the overall network traffic.

Key words : QoS, Distributed Virtual Environment, Multimedia Streaming, Multicast

1. 서 론

1.1 연구 배경

가상공간이 또 하나의 현실 세계로 인식되어 사용자가 참여 가능한 가상공간에 대한 요구가 증가하고 있으나 아직까지 다수의 사용자가 공유하고 사용자간에 상호 작용이 가능한 가상현실 지원 시스템이 미비한 실정이다. 가상현실을 구현하는 경우 많은 문제에 부딪치게 된다. 우선 클라이언트간의 동기화 문제이다. 클라이언트-서버 모델의 경우 데이타가 서버를 거쳐 클라이언트로 재전송 되므로 네트워크 지연이 증가하게 된다. 피어-투-피어 방식은 서버를 거치지 않고 데이타가 각 클라이언트로 직접 전달되기 때문에 네트워크 지연 문제는 줄일 수 있으나 각 클라이언트간의 동기화를 유지하기 어려운 단점이 있다. 두 번째로는 네트워크 트래픽 증가이다. 가상 환경에 참여하는 클라이언트의 수가 증가하

· 본 연구는 1999-2001년 한국과학재단의 특정기초연구(R01-1999-00(지원을 받아 수행되었습니다.
† 비 회 원 : 한양대학교 전자통신전파공학과
jessekim@ece.hanyang.ac.kr
bekim@samsung.co.kr
†† 종신회원 : 한양대학교 전자전기컴퓨터공학부 교수
yjwon@ece.hanyang.ac.kr
park@hyuee.hanyang.ac.kr
††† 정 회 원 : 한양대학교 전자전기컴퓨터공학부 교수
jipark@hanyang.ac.kr
논문접수 : 2003년 3월 17일
심사완료 : 2003년 8월 11일

면서 전달되는 메시지의 빈도가 증가하거나 메시지의 데이타량이 커지기 때문이다. 이 경우 데이타 손실로 인한 패킷 압축 또는 여러 개의 패킷을 모아 하나의 패킷으로 보내는 패킷 통합, 그리고 패킷 전송 횟수를 줄이는 방법 등을 사용한다. 또한 멀티캐스트 프로토콜을 사용하여 네트워크 트래픽을 줄일 수 있다. 본 논문에서는 특히 클라이언트로 전송되는 데이타량을 줄이기 위해서 클라이언트의 QoS에 따라서 스트림을 트랜스코딩 하는 기법에 중점을 두고 있으며 멀티캐스트 프로토콜을 사용하여 클라이언트의 수가 증가하여도 네트워크 트래픽을 효율적으로 줄일 수 있는 방안을 제시한다. 본 논문에서 중점적으로 다루고자 하는 분산 가상 시스템의 특징을 살펴보면 다음과 같다.

- 다수의 사용자가 참여하며 동일한 가상공간을 공유한다.
- 각 참여자들은 실영상 스트림 또는 3D 아바타를 통해서 동적인 상호 작용이 가능하다.
- 각 참여자들의 상호 작용에 따라서 서버와 네트워크의 리소스를 적응적으로 조절하여 시스템의 부하를 최소화하여야 한다.
- 가상공간에서 3D 그래픽과 스트림 미디어, 오디오, 텍스트 등의 다양한 미디어 타입을 지원해야 한다.

1.2 관련 연구

DIVE 시스템[1-4]은 다수의 사용자와 어플리케이션이 가상환경에서 서로 공유하며 실시간으로 상호교환 및 통신을 할 수 있는 시스템이다. DIVE 프로젝트는 대규모 분산 가상현실을 지원하기 위해서 데이타 교환에 의해 발생하는 네트워크 지연을 최소화하는데 주안점을 두었다. 이 시스템의 장점은 참가자들이 독립적으로 항해하고 다양한 서비스와 툴들을 사용하는 환경을 제공하여 오디오와 비디오 기능들을 사용할 수 있다. DIVE에서는 멀티캐스트를 이용한 피어-투-피어 방식을 사용하여 데이타를 전송한다. 또한 손실된 패킷의 재요청 및 재전송으로 인한 오버헤드를 줄이기 위해 오브젝트의 직선 속도 이동에 기반한 간단한 dead-reckoning 알고리즘을 제안하였다. 반면 단점으로 dead-reckoning 알고리즘은 단순한 직선 운동을 하는 오브젝트에는 적합하나 복잡한 모션에는 적용되지 않는다.

NPSNET[5-7]은 국방 및 오락물을 목표로 대규모 가상현실을 구현하기 위한 인간-컴퓨터 상호작용과 소프트웨어 기술에 바탕을 두었다. NPSNET은 네트워크 가상 환경, 컴퓨터 자체적 형성, 인간-컴퓨터 상호작용, 국방 및 오락물 협력 등 다양한 연구분야에 초점을 맞추고 있다. NPSNET은 아바타가 자신의 위치와 인접한 지역의 데이타에 관심을 가진다는 사실에 근거하여 가상공간을 육각 셀로 분할하고 각 셀을 하나의 멀티캐스트 그룹에 대응시키는 방법을 사용한다. 아바타는 자신

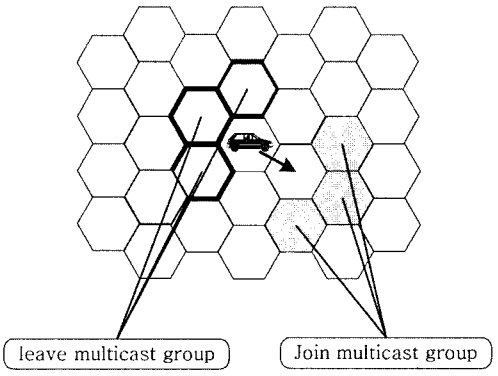


그림 1 NPSNET의 분산 가상공간 분할 방법

의 지역을 포함하여 인접한 지역, 즉 모두 7개의 멀티캐스트 그룹으로부터 정보를 받아서 아바타가 가상공간을 움직일 때 동일한 개수의 멀티캐스트 그룹에 참여 또는 떠나게 된다. 그림 1은 아바타의 이동에 의해 참여하거나 떠나는 멀티캐스트 그룹을 보여주고 있다.

NPSNET의 장점은 사용하는 지역에 따른 멀티캐스트 그룹 할당하여 각 아바타 사이의 전송되는 데이타의 양을 특정 지역으로 제한함으로써 호스트의 프로세싱 부하를 줄일 수 있을 뿐 아니라 네트워크를 통해 전송되는 트래픽 양을 효과적으로 제어할 수 있다는 것이다. 반면 단점으로는 논리적 지역 분할과 실제 가상 환경에서의 의미적인 지역 분할이 일치하지 않는 경우 오브젝트의 이동에 따른 멀티 캐스트 그룹의 잦은 변경으로 네트워크 로드가 증가하게 된다. NPSNET의 가상공간을 육각 셀로 분할하는 방법은 아바타의 지역성에 바탕을 두고 있다. 즉 오브젝트의 관심 영역이 무한하지 않기 때문에 오브젝트가 위치한 지역을 중심으로 주위의 영역에서만 데이타를 받는 방식을 선택하였다.

MASSIVE[8-12]는 분산 다중 사용자 가상현실 시스템으로 다수의 사용자가 3차원 그래픽과 실시간 패킷 오디오를 통해 통신한다. MASSIVE의 핵심 기술은 focus와 nimbus의 개념이다. MASSIVE에서는 focus와 nimbus에 따른 아바타 사이의 상호 인지 단계를 결정한다. 여기서 focus는 관찰자의 관심 영역이며 nimbus는 소스의 영향 범위를 의미한다. NPSNET이 아바타의 지역성에 따라 제한적인 데이타를 전송 받는 방법을 사용한데 반하여 MASSIVE에서는 아바타의 관심 영역과 영향 범위로 사용자의 인지를 조절하여 전송되는 데이타량을 조절하는 좀 더 유연한 QoS 모델을 제안하였다. 오브젝트의 focus, nimbus에 따른 상호 인지를 결정하는 방법은 각 오브젝트가 서로의 focus, nimbus내에 있을 때 가장 우수한 QoS를 가지며 한 쪽이 다른 한쪽에 속하는 경우는 주변 인지를 가지고 각 오브젝트의

focus, nimbus가 아무 관계없을 때는 최소의 인지를 가지게 된다. 이러한 상호 인지 단계에 따라서 대상 매체 또는 아바타의 QoS가 조절되는데 예를 들면, 오디오 볼륨이나 비디오의 품질이 변하게 된다. MASSIVE의 장점은 새로운 미디어가 핵심 시스템에 영향을 주지 않고 쉽게 추가될 수 있다는 것이다. MASSIVE 에서 제안한 focus, nimbus에 따른 상호 인지 모델은 1인칭 시점의 가상공간에 맞게 설계되었기 때문에 자신의 아바타가 화면상에 보이지 않는, 즉 자신의 아바타의 시점과 뷰어의 시점이 일치하게 된다. 따라서 3인칭 시점의 가상공간에 대해서는 이 개념을 그대로 적용할 수 없다. 또한 각 아바타는 3D 도형 위에 사람의 얼굴만을 비디오 스트림으로 표현하므로 아바타간의 동적인 상호 작용이 이루어지기 어렵다. 즉 정적인 아바타 모델만을 정의하였을 뿐 동적인 아바타에 대한 모델이 정의되지 않았고 이에 대한 QoS도 제시되지 않았다.

분할 알고리즘(partitioning algorithm)을 제안한 분산 가상현실 시스템[13]은 사용자가 가상현실공간에서 3차원 객체인 아바타로 표현된다. 각 아바타는 가상공간의 사무실을 오가며 실시간으로 상호동작한다. 사용자는 사무실에서 참가자의 신용상태 등을 조회할 수 있고, 내장된 데이타베이스를 통해 질의를 하며 정보를 공유한다. DVE 시스템에서의 고질적인 문제점인 범위성 문제는 클라이언트의 수가 증가할 때 클라이언트들과 DVE 서버들이 발생하는 네트워크 트래픽의 양이 크기 때문에 발생한다. 이러한 문제점을 여러 대의 서버를 할당하여 클라이언트의 요구를 적절히 분산시켜 해결한다. 이 시스템에서 접근한 방법은 분할 알고리즘을 사용하여 여러 대의 서버에 적절하게 분할하여 범위성 문제를 해결하였다. 분할 알고리즘은 선형 최적화 기법에 근거하여 수치적으로 효율적이라는 것을 증명하였다. 이 알고리즘의 장점은 서버의 작업량을 동등하게 분할하고 동시에 통신부하를 감소시킬 수 있다. 분할 알고리즘의 분산가상현실 시스템은 QoS를 사용하지 않고 분할 기법을 사용하여 네트워크의 부하를 해결하였다. 하지만 이 시스템은 실영상을 지원하지 않으며 단지 3차원 그래픽을 사용하여 동적인 상호교환만 하는 단점을 갖고 있다.

DSG(Destination Set Grouping)방법[14]은 Replicated 스트림 방식을 사용한 멀티캐스트 분산 시스템이다. Replicated 스트림 방식은 클라이언트의 네트워크 자원에 따라 비디오 스트림의 품질을 조절하기 위한 피드백 제어 메시지를 사용한다. 클라이언트는 받고 있는 비디오의 품질을 평가하여(예를 들어, 패킷 손실율) 피드백 메시지를 서버측에 전달하고 서버는 이에 따라 압축 매개변수를 달리하여 스트림을 전송한다. 멀티캐스트 환경에서 다수의 클라이언트가 참가 할 때, 모든(또는 대부분의) 클라이언트들이 동시에 피드백 메시지를 서버측에 보낼 경우 일어날 수 있는 문제점과 클라이언트의 네트워크 수용력에 동등한 품질의 비디오 스트림을 받아야 한다는 공정성에 관한 문제를 DSG 방법을 통하여 해결하고 있다. DSG에서 전송측은 몇 개의 같은 내용의 비디오 스트림을 유지하지만, 서로 다른 네트워크 품질을 가지고 있다. 각 스트림은 클라이언트들의 그룹에 의해 피드백으로 제어되고, 클라이언트들은 네트워크 상황에 따라 그룹들 사이에서 이동하게 되어 공정성을 향상시킬 수 있다. 즉 클라이언트의 연결상황에 따라 동적으로 품질을 조절한다. Replicated 스트림 방식의 장점은 비디오의 품질을 평가하여 네트워크 상황에 맞춰 알맞은 품질의 스트림을 선택한다는 것이다. 반면, 본 논문에서는 하나의 클라이언트가 여러 개의 스트림을 받게 되므로 중요하지 않은 스트림의 경우 저품질의 스트림을 받게 하여 주어진 네트워크 자원에서 총체적인 메시지의 수를 줄이는데 목적이 있다. 따라서, 많은 수의 클라이언트가 참가하였을 때, 한 클라이언트에서 받아야 할 스트림의 개수는 많아지기 때문에 제한된 네트워크 자원에서 이를 모두 수용하기란 어렵다.

가상 댄싱 환경에서 다수의 분산 클라이언트들에게 데이타를 전송할 때 데이타의 양에 따른 네트워크 부하가 발생하게 된다. 특히 실영상을 전송하는 경우 모든 사용자가 다른 사용자의 실영상 스트림을 모두 전송 받아야 하기 때문에 이에 따른 네트워크 및 렌더링 부하는 더욱 커지게 된다. 가상현실 환경(특히 혼재 현실)에서 데이타 교환에 따른 네트워크 트래픽 부하의 해결을 위하여 기존의 가상 환경들은 여러 가지 방법을 사용하여 이를 해결하고 있다. 대부분의 멀티캐스트 그룹을 사용하는 가상환경들은 아바타들이 출현한 가상공간에서 단지 간단한 움직임만을 표현하고 동적인 움직임에 대한 품질의 조절을 하지 못한다는 단점이 있다. 또한 가상공간에서 다수의 참가자가 동시에 스트림을 전송하였을 때 각 사용자에게 전송되는 수 많은 영상들의 처리를 효율적으로 처리 할 수 있는 부분이 미비한 상태이다. 다수의 사용자가 참가할 때 야기될 수 있는 네트워크 부하를 본 논문은 사용자가 밀집한 가상 댄싱 환경에서 사용자간의 거리에 따라 QoS 단계를 나누어 별도의 멀티캐스트 채널을 사용함으로써, 필요한 네트워크 자원과 메시지 수를 줄여 보다 원활하고 몰입감 있는 가상 환경을 제공하는 것을 목표로 하고 있다. 본 논문에서는 제한적인 3인칭 시점의 가상공간을 구성하고 뷰의 허용거리를 정의하여 이에 따라 가상공간에서 보여지는 아바타들의 화질을 조절하는 Visual QoS와 아바타들 사이의 영향 범위 및 아바타간의 거리에 의해 사용자들에게 들리는 소리의 크기를 조절하는 Audio QoS

의 계층적 QoS 모델을 제시하였다.

본 논문의 구성은 다음과 같다. 2장에서는 TIE 시스템에 대한 전체적인 개요에 대해서 설명하고 3장에서는 가상 댄싱 공간에 적합한 계층적 QoS 모델을 Visual QoS와 Audio QoS로 나누어 소개한다. 4장에서는 TIE 시스템의 구조를 설명하고 5장에서는 실험을 통해 TIE 시스템의 성능분석과 그 결과를 살펴본다. 6장에서는 QoS 모델의 신빙성과 정확성을 위하여 가상댄싱 공간의 시뮬레이션을 제작하여 그 결과를 살펴본다.

2. TIE 시스템 개요

TIE(Tele-Immersion/Emersion)는 원격지의 사용자들이 마치 한 공간에 있는 것처럼 가상 세계에 출현(Emersion)하여 서로 상호 작용 하면서 커뮤니케이션 할 수 있고 동시에 사용자의 몰입감(Immersion)을 증가시키는 가상공간 시스템을 구현하는 것이다. 이러한 시스템을 구현하기 위해서 실영상과 3D CG 아바타가 참여하는 가상 댄싱 공간을 설계하였다. 먼저 시스템 부분에서는 사용자의 상호 작용 정도에 따른 계층적 QoS 구조를 제안하였고, 단말 부분에서는 사용자의 실영상과 3D 그래픽이 혼재된 가상공간 렌더링 기법을 개발하였다. 네트워크 부분에서는 적응적인 QoS를 기반으로 한 멀티캐스트 라우팅 프로토콜 설계하여 액티브 라우터 실험 망에서 운용 가능성을 검증하였다.

그림 2는 본 논문에서 구현한 가상 댄싱 공간 개념도를 나타낸다. 가상 댄싱 공간은 클라이언트-서버 구조로 각 사용자는 CG 아바타 또는 실영상 아바타로 가상 댄싱 공간에 참여한다. 모든 사용자는 동일한 가상 공간을 공유하지만 아바타의 위치 이동에 따라 사용자가 관찰하는 가상 공간의 범위가 다르게 된다. 예를 들어 사용자 A의 입장에서 보면 아바타 A가 가장 가까운 위치에 있고 아바타 B는 보이지 않는다. 그러나 사용자 B의 입장에서 보면 아바타 B가 가장 가까운 위치에 있고 아바타 A는 보이지 않는다. 즉 사용자 별로 보게 되는 아바타들의 위치가(거리와 각도) 다르고 사람의 시야 특성상 먼 거리보다 가까운 거리에 있는 사물이 크고 또렷하게 보인다. 이러한 특성을 분산 가상 댄싱 공간에 적용하여 사용자의 시점에 상대적인 아바타의 위치에 따라 가상 공간에 참여하는 아바타의 화질을 달리하였다. 가상 공간에 실영상의 아바타가 참여하는 경우 프레임 레이트, 프레임 크기, 인코딩 방법을 다르게 하여 아바타의 화질을 조절할 수 있다. 즉 서버에서는 사용자의 시점에서 본 아바타들의 위치에 따라 아바타의 QoS 단계를 결정하고, 이 값에 의해 사용자에게 전송할 실영상을 트랜스코딩한다. 본 논문에서 제안하는 계층적 QoS 구조를 분산 가상 댄싱 공간에 적용하여 데이타가 전송되는 목적지의 개수와 네트워크를 통해 전송되는 데이타의 크기를 줄이고 가상 댄싱 공간을 렌더링하는 데 필요한 단말의 리소스를 감소시키는 것을 목적으로 한다.

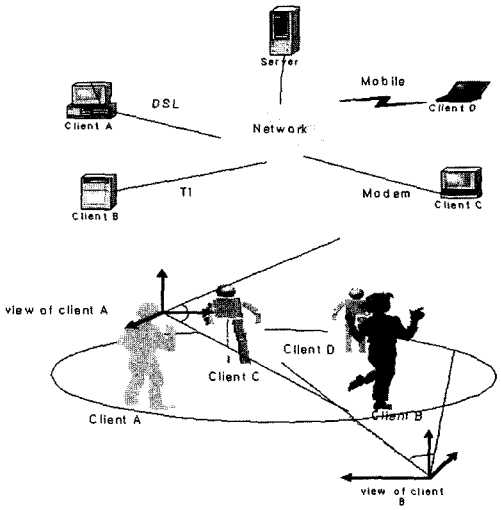


그림 2 가상 댄싱 공간 개념도

3. TIE 시스템의 계층적 QoS 모델

분산 가상 시스템인 TIE 시스템을 효율적으로 구현하기 위해서는 분산 가상 시스템을 구현하는데 필요한 리소스의 양을 줄일 수 있어야 한다. 식 1의Information Principle[15]에서 보여지듯이 5개의 변수를 조절하여 분산 가상 시스템에서 요구되는 리소스 양을 줄일 수 있다.

$$Resources \approx M \times H \times B \times T \times P \quad (1)$$

- M = 네트워크를 통해 전송되는 메시지의 개수
- H = 메시지가 전달되는 평균 호스트의 수
- B = 메시지를 전송하는 데 필요한 평균 네트워크 대역폭 크기
- T = 패킷이 목적지까지 전송되는데 요구되는 제한 시간
- P = 메시지를 처리하는 데 필요한 프로세서 사이클의 개수

구현하고자 하는 분산 가상 댄싱 공간은 다음과 같은 특징을 가진다.

- 가상 댄싱 공간 내 아바타의 밀집도가 높다.
- 실영상과 CG 아바타 정보를 실시간으로 전송하므로 처리해야 할 전송 데이타량이 많다.
- 직접 제어 할 수 있는 동적인 아바타를 사용하여 아바타간에 다양한 상호 작용을 지원한다.

특히 CG영상과 실영상이 혼재된 가상 공간이므로 처리해야 할 스트림 데이타가 많으며 이 스트림을 렌더링

하기 위해서 많은 리소스를 사용한다. 네트워크를 통해 전송되는 데이타량을 줄이기 위해서 아바타간의 상호작용에 따라 계층적으로 QoS를 할당하여 각 클라이언트에게 멀티캐스트 한다. 이 방법을 Information Principle에서 살펴보면 H와 B를 줄일 수 있으나 반면에 M과 P가 다소 증가하게 된다. 다음 절에서는 메시지가 전달되는 평균 호스트 수와 메시지를 전송하는데 요구되는 평균 네트워크 대역폭을 줄이기 위해 제안한 계층적 QoS 구조를 설명할 것이다.

3.1 기본 정의

본 연구에서 사용하는 몇 가지 개념을 다음과 같이 정의한다. 먼저 *아바타*는 가상 댄싱 공간에 참여하는 사용자의 *대리인*을 말한다. 가상 댄싱 공간에 참여하는 사용자의 아바타를 *로컬 아바타*(local avatar)라고 하며 다른 사용자의 아바타를 *리모트 아바타*(remote avatar)라고 한다. 또한 아바타의 시점과 관찰자의 시점이 동일한 경우를 *1인칭시점*이라 하고, 동일하지 않은 경우를 *3인칭시점*이라고 나타낸다. 여기서 *관찰자*는 가상 댄싱 공간에서 *뷰어*를 말하며 사용자는 관찰자의 시점으로 가상공간을 보게 된다. 1인칭 시점에서 관찰자는 로컬 아바타를 볼 수 없지만 3인칭 시점인 경우 관찰자는 제3자의 시점으로 로컬 아바타를 보면서 아바타를 임의의 위치로 이동시키거나 아바타의 움직임을 제어할 수 있다. 1인칭 가상 공간에서는 상대적으로 몰입감을 더 줄 수 있으나 사용자 본인의 아바타인 로컬 아바타를 볼 수 없으므로 움직임에 있어 제한적이다. 따라서 1인칭 가상 공간 보다 3인칭 가상 공간에서 아바타간의 다양한 상호 작용이 이루어지게 된다. 본 연구에서는 제한적인 3인칭 시점의 가상 댄싱 공간을 구현하였다. 본 연구에서 정의한 제한적인 3인칭 시점이란 관찰자의 위치에 의해 로컬 아바타가 이동 할 수 있는 거리가 제한되는 3인칭 시점을 말한다. 여기서 아바타가 이동할 수 있는 거리를 *뷰의 허용거리*라고 한다.

3.2 Visual QoS

3인칭 시점의 가상 공간에서 사용자는 그림 3의 관찰자의 위치에서 가상 공간을 관찰하며 로컬 아바타가 되어 가상 공간에 출현한다. 사람은 시야 특성상 양안 각도 내의 한정된 영역을 볼 수 있으며, 먼 거리보다 가까운 거리에 있는 사물이 더 잘 보인다. 따라서 분산 가상 댄싱 공간에서도 관찰자는 보는 각도에 따라 제한된 영역을 관찰하고, 아바타와의 거리에 따라 원근감이 달라져야 한다. 또한 관찰자는 자신의 대리인인 로컬 아바타를 항상 관찰할 수 있어야 한다.

이러한 특징을 만족하기 위해서 본 연구에서는 제한적인 3인칭 시점의 가상 댄싱 공간을 제안하였다. 첫째 관찰자와 아바타의 거리에 따라 원근감이 다르므로 관찰자

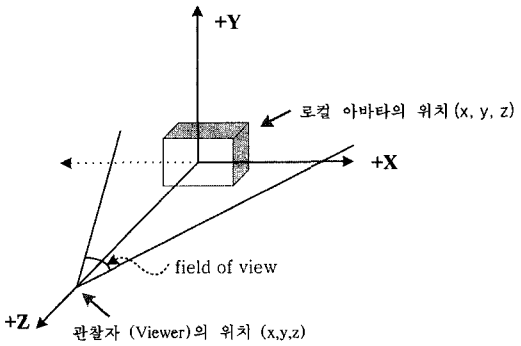


그림 3 가상 공간에서 관찰자와 아바타의 위치

가 느끼는 화질을 달리한다. 이때 관찰자가 느끼는 화질을 Visual QoS로 정의한다. 둘째 로컬 아바타의 이동 범위를 제한하여 로컬 아바타가 관찰자의 시야 내에 있으면서 동시에 로컬 아바타의 화질을 보장하는 방법을 제안한다. 즉 로컬 아바타가 뷰의 허용 범위를 벗어났을 경우 관찰자의 위치를 이동시키는 방법을 사용한다.

CG 아바타의 경우는 Visual QoS를 만족시키기 위하여 LOD(Level of Detail)를 사용한다. LOD란 그래픽의 간략화를 여러 단계로 수행한 결과로 상세함이 다른 여러 단계의 메쉬들을 얻는 것을 말한다. 즉, 시각적 효과를 최대한 유지하는 가운데 아바타의 특성, 거리 등에 따라 LOD를 변화시켜 최적의 렌더링을 한다.

3.2.1 관찰자로부터 아바타까지의 거리 계산

가상댄싱공간에서 아바타의 위치를 $(x_\alpha, y_\alpha, z_\alpha)$라고 하고 관찰자의 위치를 $(x_\beta, y_\beta, z_\beta)$라고 하면 관찰자로부터 아바타까지의 거리 D는 식 2와 같이 나타낸다.

$$D = |z_\beta - z_\alpha| \tag{2}$$

관찰자의 시야 특성에 따라 아바타의 화질은 가상공간에서의 관찰자로부터 아바타까지의 거리 D에 반비례한다. 즉 관찰자로부터 아바타까지의 거리 D가 가까울수록 아바타의 화질은 우수하며 D가 커질수록 아바타의 화질은 감소하게 된다.

3.2.2 Visual QoS 단계 결정

아바타와 뷰어간의 최대 거리 D_{max}와 최소 거리 D_{min}는 가상 댄싱공간의 댄싱플로워 평면의 크기에 의해 구할 수 있다. D_{min}은 아바타와 뷰어의 위치가 일치할 때 얻어지는 값이 되므로 0이 된다. D_{max}는 댄싱플로워 평면의 가로의 길이 x와 세로의 길이 y에서 최대 거리이고 이는 식 3에서 정의한다.

$$D_{max} = \sqrt{(x_{max} - x_{min})^2 + (y_{max} - y_{min})^2} \tag{3}$$

따라서 거리 D에 의해 아바타의 화질을 결정하는 QoS 매핑 곡선을 식 4에서 정의한다.

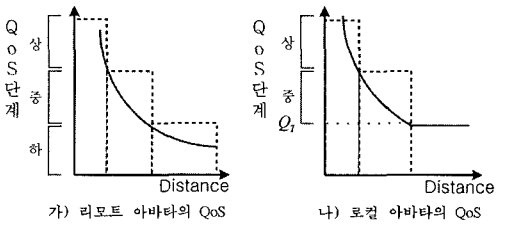
$$\text{QoS mapping curve} = k \times \frac{1}{D^{1/c}}, \text{ where } D_{\min} \le D \le D_{\max} \quad (4)$$

QoS 매핑 곡선은 다음의 특징을 가진다. 첫째 식 4에서와 같이 관찰자로부터 아바타까지의 거리 D를 계산하여 QoS 매핑곡선에 반비례하게 각 아바타의 QoS 단계를 결정한다. QoS 단계의 설정은 QoS 매핑 곡선에 해당하는 최대값과 최소값을 QoS 단계의 개수로 나누게 되면 뷰어와 아바타간의 거리를 따져 QoS 값을 설정할 수 있다. 여기서 k는 QoS단계가 할당되는 비율을 조절하는 상수이고 c는 QoS 매핑 곡선의 기울기를 조절하는 상수이다. 즉, c의 값을 2, 3, 4로 조절하여 매핑 곡선의 기울기를 조절할 수 있는데 이것은 가상 댄싱공간에 참여한 실영상 아바타의 수와 댄싱플로워 면적 등을 고려하여 설정한다.

아바타의 QoS는 [상 중 하] 세 단계로 두었으며 여기서 결정된 QoS 단계에 의해 아바타의 화질이 결정된다. 실영상 아바타의 경우 실영상 데이타의 메시지 수를 줄이는 방법으로 프레임 레이트, 영상의 크기, 비트 레이트 등을 조절하는 방법을 사용할 수 있다. 본 논문에서는 Visual QoS를 만족시키기 위한 방법으로 프레임 레이트를 조절하는 방법을 선택하였다. RTP를 이용하여 스트림을 전송하는 경우 JMF에서 제공하는 여러 코덱 중 우리는 H.263을 사용하였다. H.263을 선택한 이유는 낮은 프레임 레이트에서 매우 낮은 비트 레이트로 고화질의 영상을 코딩하고 PSTN, 이동통신 네트워크, narrowband ISDN 등도 잘 지원하기 때문이다. 실영상 아바타의 경우 QoS [상 중 하]단계에 따라 비디오 스트림의 프레임 레이트를 [15Hz, 10Hz, 5Hz]로 변경하여 트랜스코딩하며, 3D 아바타는 아바타의 LOD를 변경한다. 따라서 관찰자의 시야 특성에 따라 잘 안 보이는 아바타의 화질을 낮추어 전송하고 잘 보이는 아바타의 화질을 높게 하여 전송하므로, 가상 댄싱 공간을 관찰하는 관찰자의 Visual QoS를 손상하지 않고도 네트워크 대역폭과 단말에서 렌더링 하는데 필요한 시스템 리소스를 줄일 수 있다.

둘째, 그림 4와 같이 로컬 아바타와 리모트 아바타는 다른 QoS 매핑 곡선을 사용한다. 제한적인 3인칭 시점의 가상 댄싱 공간에서 로컬 아바타는 뷰의 허용 거리 내에서 이동하게 된다. 즉 로컬 아바타가 뷰의 허용 거리를 벗어났을 경우 관찰자의 위치를 변경하여 로컬 아바타가 뷰의 허용 거리 내에 들어오도록 한다. 그 이유는 관찰자에게는 로컬 아바타가 가상 댄싱 공간의 주인공이기 때문에 자신의 아바타를 계속 관찰하면서 제어할 수 있어야 하기 때문이다. 로컬 아바타가 최대 뷰의 허용 거리에 위치했을 때 QoS 단계 값을 임계값 Q_T, 로 나타낸다. 로컬 아바타는 Q_T 일 때 가장 화질이 떨어지며 QoS 매핑 곡선에 의해 로컬 아바타의 QoS를 항상 Q_T 이상으로 보장할 수 있다. 로컬 아바타의 QoS를 계산하는 식은 식 5에 나타난다. 여기서 QM은 리모트 아바타의 QoS 매핑 곡선이며 QM_L은 로컬 아바타의 QoS 매핑 곡선이다.

$$QM_L = \max(Q_T, QM) \quad (5)$$



가) 리모트 아바타의 QoS 나) 로컬 아바타의 QoS

그림 4 QoS 매핑 곡선

셋째 QoS 매핑 곡선 특징에 의해 각 QoS 단계가 결정될 확률이 다르다. 그림 4에 의하면 QoS 매핑 곡선은 기울기가 급격하게 감소하다 점점 줄어드는 형태를 가지므로 아바타의 QoS 단계가 '상'으로 결정될 확률이 가장 적다. 이 QoS 매핑 곡선에 의하면 대부분의 아바타의 QoS가 단계가 '중' 이하로 결정되어 단말에서 렌더링에 따른 리소스 사용량을 줄일 수 있다.

3.2.3 계층적 QoS 전송

동일한 가상공간을 다수의 사용자가 공유하고 있으나 각 관찰자의 시점이 다르므로 이에 따라 아바타의 화질이 다르게 결정된다. 예를 들어 A 사용자의 로컬 아바타와 근접한 거리에 있는 아바타라 할지라도 B 사용자에게는 보이지 않을 수도 있다. 따라서 서버에서는 각 사용자별로 가상 댄싱 공간에 참여하는 모든 아바타의 QoS 단계를 계산해야 한다.

각 사용자에 따른 아바타의 QoS 단계를 $m \times n$ QoS 매트릭스에 저장한다. 여기서 m은 아바타의 수이며 n은 사용자의 수이다. 이 가상 댄싱 공간에서는 한 사용자가 하나의 로컬 아바타를 생성하므로 사용자와 아바타의 개수가 같다. 그림 5는 $m \times n$ QoS 매트릭스이다. 가상 댄싱 공간에서 k번째 아바타의 이동에 의하여 각 관찰자가 보는 k번째 아바타의 위치가 달라진다. 따라서 서버에서는 가상 댄싱 공간에 참여하는 모든 사용자 $(1, 2, 3, \ldots, j)$에 대해 k번째 아바타의 QoS 단계 $(Q_{k1}, Q_{k2}, Q_{k3}, \ldots, Q_{kj})$를 계산하여 QoS 매트릭스에 업데이트 한다. 서버는 아바타의 스트림을 QoS 단계에 따라 각각의 멀티캐스트 그룹으로 멀티캐스트 한다. 또한 서버에서는 k관찰자가 보는 QoS 단계

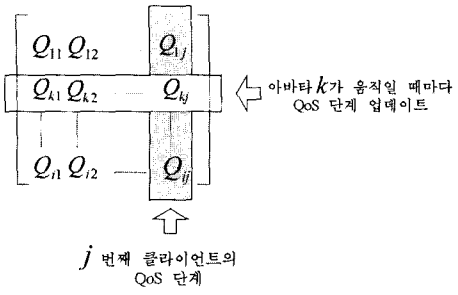


그림 5 $m \times n$ QoS Matrix, $(m=n)$

$(Q_{1k}, Q_{2k}, Q_{3k}, \ldots, Q_{ik})$에 해당하는 멀티캐스트 정보를 k사용자에게 전송한다. 따라서 단말에서는 서버로부터 조인해야 할 멀티캐스트 그룹 정보를 받고 해당 멀티캐스트 그룹에 조인하여 관찰자의 시점에 의한 가상 댄싱 공간을 렌더링 할 수 있다. 즉 가상 댄싱 공간에서 관찰자의 시점에 의해 아바타의 화질이 떨어지는 경우 화질이 낮은 스트림을 받고 아바타의 화질이 높은 경우 화질이 높은 스트림을 전송 받기 때문에 네트워크 대역폭을 줄이면서 사용자의 Visual QoS를 만족시킬 수 있다.

3.3 Audio QoS

가상 댄싱 공간은 아바타간의 다양한 상호 작용을 지원한다. 가상 댄싱 공간에서 아바타는 댄싱 음악에 맞추어 춤을 추기도 하고 다른 아바타와 대화를 하기도 한다. 가상 댄싱 공간에서 실제로 말을 하거나 소리를 듣는 주체는 로컬 아바타가 아니라 사용자이므로 사용자가 느끼는 소리의 좋고 나쁨을 Audio QoS의 좋고 나쁨으로 나타낸다. 이때 로컬 아바타가 가상 댄싱 공간 내에 참여하는 모든 아바타나 오브젝트의 소리를 들을 수 없으므로 가상 댄싱 공간에서 소리가 전달되는 영향 범위인 아바타의 영향 범위를 정의하였다. 따라서 가상 댄싱 공간에서는 아바타의 영향 범위와 아바타간의 거리에 의해 사용자에게 들리는 소리의 크기를 결정한다.

3.3.1 아바타의 영향 범위

아바타의 영향 범위는 아바타나 다른 오브젝트가 가상 댄싱 공간 내에서 영향을 미치는 범위이다. 엄밀하게 정의하면 아바타 상호 작용의 영향 범위로서 아바타의 행동이 다른 아바타에게 미칠 수 있는 영향 범위이다. 예를 들어 총을 가진 사람과 무기가 없는 사람의 영향 범위가 다르다. 왜냐하면 총을 가진 사람은 사격 거리 내에 있는 모든 아바타에게 영향을 줄 수 있지만 무기가 없는 사람은 자신의 손발이 닿는 범위에서만 영향을 줄 수 있기 때문이다.

구현한 가상 댄싱 공간에서 아바타의 영향 범위는 아바타의 종류(로컬, 리모트), 매체(비디오, 오디오), 특성(댄싱 음악, 아바타 목소리), 접속환경에 따라 다르게 정의한다. 사용자는 자신과 로컬 아바타를 동일시하기 때문에 로컬 아바타의 영향 범위를 리모트 아바타의 영향 범위보다 크게 한다. 또한 아바타의 영향범위는 관찰자로부터 아바타의 거리에 따라 적응적으로 변한다. 아바타가 관찰자로부터 멀어지면 아바타의 화질이 감소하고 크기가 작아지므로 이에 따라 아바타의 영향 범위도 줄어들게 된다. 즉 아바타의 영향 범위는 관찰자로부터 아바타까지의 거리에 반비례하며 아바타의 영향 범위는 중심의 좌표와 반경으로 나타낸다. 또한 아바타의 목소리도 아바타의 영향 범위 내에 있는 아바타의 목소리만 듣게 되어 참가한 모든 아바타의 목소리를 듣는 것 보다 몰입감을 더해줄 수 있다.

3.3.2 Audio QoS 결정

오디오 QoS는 아바타의 영향 범위와 아바타간의 거리에 의해 결정된다. 식 6에서와 같이 아바타의 영향 범위가 겹치거나 접할 경우 아바타의 대화가 상대측 아바타에게 전달되며 아바타의 영향 범위가 만나지 않을 경우 상대방측 아바타의 소리가 들리지 않는다. 아바타 k의 영향 범위의 반지름 R_k는 접속환경에 따라 고속망에 접속한 사용자는 그 길이가 넓고, 56Kbps 모뎀 등으로 접속한 사용자는 상대적으로 네트워크 대역폭이 좁기 때문에 그 길이가 짧다. 56Kbps 모뎀의 경우 네트워크 대역폭의 제한으로 인하여 비디오 아바타는 선택할 수 없고 CG 아바타만 선택이 가능하다. 또한 이러한 제한으로 인하여 오디오 서비스도 받을 수 없다. 다른 접속환경은 허용된 대역폭에 따라 ADSL의 경우 4Mbps, LAN의 경우 T1, T3, E1등 다양하고 속도의 편차가 크기 때문에 하향속도를 3Mbps의 속도값으로 고정하였다. 이 속도값을 사용하여 사용자가 연결된 속도에 따라서 영향 범위의 반지름 R_k는 결정된다.

R_k : 아바타 k의 영향 반지름

d : a 아바타와 b 아바타 사이의 거리,

$R_a + R_b \geq d$: 소리가 들림

$R_a + R_b < d$: 소리가 안들림 (6)

아바타의 위치 이동과 아바타 사이의 거리에 따라 사용자에게 소리가 들리기도 하고 들리지 않기도 하므로 가상 댄싱 공간에 참여하는 사용자는 아바타와 일체감을 느낄 수 있다.

3.4 QoS Multicasting

클라이언트로부터 새로운 스트림이 들어오게 되면 서버는 각 클라이언트의 QoS 정보를 업데이트하고 QoS 단계의 개수만큼 이 스트림의 데이타 소스를 복제하여 트랜스코딩하고 각각 다른 멀티캐스트 그룹으로 멀티캐스팅 된다.

QoS에 따른 총 멀티캐스트 채널의 수는 아바타의 수 × QoS 단계의 개수이다. 따라서 각 클라이언트는 QoS

단계의 개수만큼의 채널이 형성되며 하나의 멀티캐스트 그룹은 QoS 단계의 개수만큼 채널이 형성되어 멀티캐스팅 되고 각 채널은 서로 다른 멀티캐스트 주소를 사용한다. 클라이언트는 QoS 정보에 따라 알맞은 멀티캐스트 그룹의 채널로 참가하여 해당 스트림을 받게 된다. 멀티캐스트 채널의 수는 동적으로 변한다. 예를 들어, 네 명의 사용자가 참가한 가상공간에 다른 한명이 추가적으로 참가를 한다면(QoS 단계가 3일 경우) 멀티캐스트 채널의 수는 12에서 15가 된다. 이와 반대로 참가했던 사용자가 가상공간을 떠날 경우 떠난 사용자의 채널(3개)은 없어지게 된다. 멀티캐스트 채널은 멀티캐스트 주소범위 D 클래스에서 233.0.0.x의 주소범위를 사용하였다. 따라서 첫 번째 사용자는 233.0.0.0~233.0.0.2까지의 주소를 사용하여 RTP를 통해 미디어를 멀티캐스팅한다. 이 주소 값은 동적으로 앞에서부터 할당된다. 주소의 관리는 *AddressManager*에서 관리하는데 빈 공간에 대한 멀티캐스트 주소그룹은 빠른 검색을 위하여 B+트리로 구현하였다. 만약 첫 번째 사용자가 가상공간을 떠났다가 다시 들어왔을 때 주소를 B+트리의 형태로 검색하여 비워져 있는 멀티캐스트 채널(3개씩)을 할당하게 된다. 아바타가 실시간으로 접속, 떠남을 반복하는 경우 동적으로 멀티캐스트 채널의 수는 변하며 비워져 있는 순서대로 할당받는다.

유니캐스트는 네트워크상에서 단일 송신자와 단일 수신자간의 통신이다. 단일 송신자와 다중 수신자간의 통신인 멀티캐스트나, 또는 네트워크 상의 어떠한 송신자와 가장 가까이 있는 수신자 그룹간의 통신인 애니캐스트 등과 구별하기 위해 존재한다. 인터넷의 전송 방식은 전송에 참여하는 송신자와 수신자 관점에서 나누어 유니캐스트, 브로드캐스트, 멀티캐스트로 구분할 수 있다. 유니캐스트 전송 방식은 하나의 송신자가 다른 하나의 수신자로 데이타를 전송하는 방식으로 일반적인 인터넷 응용프로그램이 모두 유니캐스트 방식을 사용하고 있다. 브로드캐스트 전송방식은 하나의 송신자가 같은 서브네트워크 상의 모든 수신자에게 데이타를 전송하는 방식이다. 반면 멀티캐스트 전송방식은 하나 이상의 송신자들이 특정한 하나 이상의 수신자들에게 데이타를 전송하는 방식으로 인터넷 화상 회의 등의 응용에서 사용한다. 만약 그룹 통신을 위하여 다중 수신자들에게 동일한 데이타를 전송하고자 할 경우 유니캐스트 전송방식을 이용한다면 전송하고자 하는 데이타 패킷을 다수의 수신자에게 각각 여러 번 전송해야 하며, 이러한 동일한 패킷의 중복전송으로 인해 네트워크 효율이 저하된다. 또한 수신자 수가 증가할 경우 이러한 문제점은 더 커지게 된다. TIE 시스템의 경우 멀티캐스트방식을 사용하는데 그 이유는 QoS를 적용하여 아바타들의 화질을 조절하기 위함이다. 만약 유니캐스트방식을 사용한다면 모든 클라이언트가 해당하는 모든 스트림을 받아야 하기 때문에 비효율적이다. 예를 들어, 다섯 명의 사용자가 비디오 아바타를 선택했다고 가정하자. 그렇다면, 다섯 명의 사용자는 각 사용자의 스트림을 각각의 클라이언트에서 받아야 한다. 사용자들은 예를 들어 사용자1의 스트림을 서로 다른(또는 같은) 품질로 받아야 한다. 이것을 유니캐스트 방식으로 받는다면 각각의 사용자들에게 사용자 1의 스트림을 서로 달리하여 보내야 할 것이다. 하지만 멀티캐스트 방식을 사용한다면 단지 해당하는 그룹에만 참가하여 받기 때문에 효율적이다.

4. TIE 시스템 구조

4.1 TIE 시스템의 구조 및 전송과정

그림 6은 TIE 시스템의 데이타의 흐름을 나타낸 그림이다. 그림 6에서 볼 수 있듯이 전송하고자 하는 데이타의 특성에 적합한 전송 프로토콜을 사용하여 서버와 클라이언트간의 메시지를 교환한다. TCP 프로토콜은 연결 지향적인 프로토콜로서 목적지까지 데이타 전송을 보장하며 데이타가 전송되는 순서도 보장한다. 따라서 클라이언트의 세션 참여 및 종료 메시지는 TCP 프로토콜을 사용하여 전송한다.

UDP 프로토콜은 비연결 지향적인 프로토콜로서 패킷의 전송을 보장하지 않지만 TCP 프로토콜에 비해서 상대적으로 가볍고 데이타를 보다 빠르게 전송할 수 있다. 따라서 실시간으로 전송해야 하는 메시지를 전송하는데 적합하다. 분산 가상 댄싱 공간에서는 아바타의 상태정보와 이벤트 정보를 UDP 프로토콜을 이용하여 전송한다. UDP를 이용한 아바타의 상태 및 이벤트 정보의 전송은 UDP로 하기 때문에 데이타의 손실을 초래할 수 있다. 이를 해결하기 위해 TIE 시스템에서는 아바타의 상태 및 이벤트 정보를 주기적으로 업데이트(예를 들어

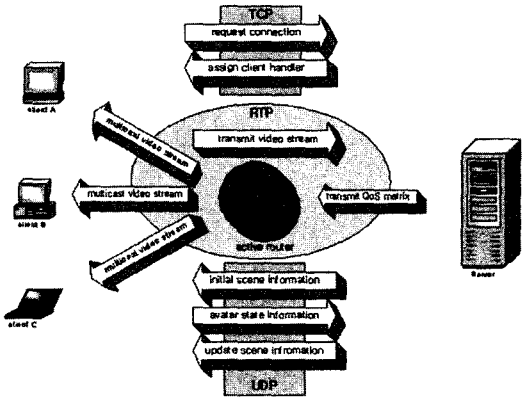


그림 6 TIE 시스템 데이타 흐름도

3초 간격)하고 있다. 이 정보들은 정보 특성상 바로 업데이트 되지 않아도 되기 때문(다른 정보에 비해서)에 데이타의 손실이 생겨도 3초안에 업데이트 되어 정보 손실에 대한 보상을 하고 있다.

RTP 프로토콜은 UDP상에서 구현된 어플리케이션 계층 프로토콜로서 부하가 적고 실시간 미디어 전송을 위해 설계되었다. TIE 시스템에서 RTP는 비디오와 오디오 스트림을 전송하는 데 사용된다.

자신의 정보를 전송하기 위해서 동영상의 경우에는 JMF에서 제공하는 RTP을 통한 H.263 코덱을 이용하여 전송하도록 설계되어 있다. 아바타의 경우에는 일반적인 네트워크 소켓을 이용하여 아바타의 이동정보나 위치정보, 움직임정보를 전송하도록 설계되었다.

4.2 TIE 시스템 서버 구조

4.2.1 계층적 QoS 구현

그림 7에서 분산 가상공간의 계층적 QoS 구조를 나타내었다. 가상공간의 장면을 관리하는 *SceneInformation*, 스트림 전송을 관리하는 *StreamHandler*, QoS를 결정하는 *QoSManager* 등의 상호작용에 의해 서버에서는 클라이언트의 QoS 단계를 결정하고 동기화된 가상 댄싱 공간을 유지한다. *SceneInformation*은 새로운 아바타가 세션에 참여하면 *AvatarInformation*를 생성하여 세션에 참여한 아바타의 정보를 관리한다. 따라서 각 클라이언트가 변경된 위치 정보를 포함한 상태 정보를 서버로 전송하면 서버에서는 이 정보로 장면을 업데이트 하여 각 클라이언트에게 변경된 장면정보를 전송한다. 아바타 A의 위치가 변경되면 각 클라이언트에서 보이는 아바타 A의 QoS도 달라지기 때문에 *QoSManager*에서는 *SceneInformation*로부터 아바타의 변경된 위치를 받아서 각 클라이언트에서 아바타 A의 QoS를 재계산 한다. 변경된 QoS는 QoS 매트릭스의 한 열에 해당된다. QoS는 <상, 중, 하>의 세 단계를 가지며 각 단계는 QoS를 나타내는 실 값으로 매핑된다. 스트림 QoS는 QoS 단계에 따라 스트림의 프레임 레이트를 변경하고 아바타 QoS는 3D 오브젝트의 LOD를 변경한다. 클라이언트로부터 새로운 스트림이 들어오면 QoS 단계의 개수만큼 이 스트림의 데이타 소스를 복제하여 트랜스코딩 한다. 세 단계의 QoS로 트랜스코딩 된 스트림은 각각 다른 멀티캐스트 그룹으로 멀티캐스팅 된다.

*StreamHandler*는 클라이언트로부터 새로운 스트림이 도착하면 *QoSManager*에서 현재 클라이언트에 해당되는 모든 아바타의 QoS 정보를 가지고 온다. 이 정보는 QoS 매트릭스의 한 행에 해당된다. 따라서 현재 클라이언트에서의 각 아바타의 QoS에 따른 멀티캐스트 그룹 정보를 액티브 라우터를 통해 클라이언트로 전송한다. 클라이언트는 이 정보를 전송 받아 멀티캐스트 그룹에 조인하여 클라이언트에게 적합한 QoS 스트림을 받을 수 있다. 따라서 각 클라이언트는 동일한 가상공간을 공유하면서도 자신에게 알맞은 QoS가 선택된 가상 댄싱 홀에 참여할 수 있다.

4.2.2 스트림 트랜스코딩

*SessionManager*는 하나의 세션을 통해서 여러 개의 스트림을 전송하거나 세션에 대한 통계 정보를 모니터링 하는 방법을 제공한다. RTP를 통해 전달되는 스트림은 *SessionManager*에 의해 세션별로 분리 되어 별

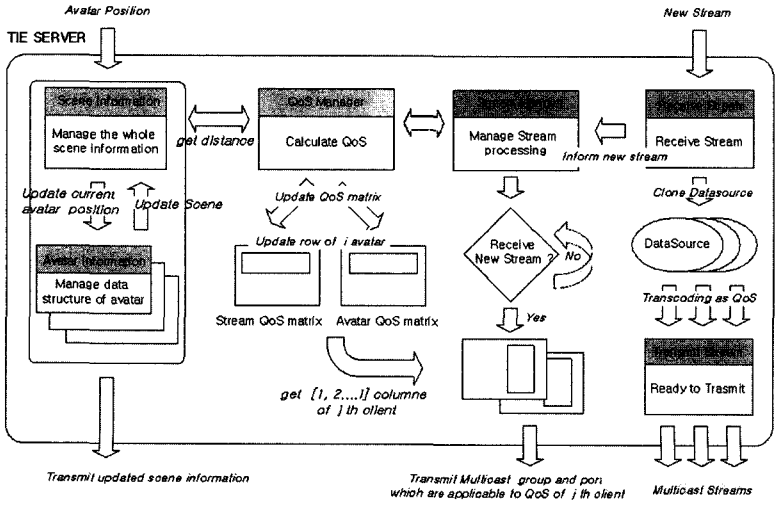


그림 7 분산 가상 공간의 계층적 QoS 구조

도의 데이타 소스로 분리된다. 즉 스트림의 오디오와 비디오는 다른 세션으로 전송되므로 별도의 데이타 소스로 저장된다. 스트림을 QoS에 따라서 트랜스코딩 한 후 클라이언트에게 멀티캐스팅 해야 한다. 따라서 입력 스트림의 데이타소스로부터 복제 가능한 데이타 소스를 생성하여 각 데이타 소스를 복제한다. 사용자 요구에 따라 스트림을 변경하기 위해서는 각 데이타 소스로부터 프로세서를 획득해야 한다. 따라서 데이타소스로부터 프로세서를 얻어 QoS에 따라 트랜스코딩 한다. 이때, 스트림 인코딩 포맷을 변경하거나 비디오 사이즈, 프레임 레이트, 스트림 압축률을 변경할 수 있는데 TIE에서는 QoS에 따라서 미디어의 프레임 레이트를 변경하여 실시간으로 인코딩 하였다. 액티브 라우터가 서버로부터 스트림을 받게 되면 이를 트랜스코딩한다. 액티브 라우터에서 H.263 코덱을 기준으로 15fps의 경우 약 90Kbps의 데이타를 5fps의 경우 약 30Kbps의 데이타를 처리하게 된다. 트랜스 코딩이 끝나면 데이타소스로부터 전송 가능한 sendstream을 얻어 스트림을 멀티캐스팅 한다.

4.3 TIE 시스템 클라이언트 구조

TIE 시스템의 클라이언트는 서버 및 액티브 라우터에 접속되어 실시간으로 정보를 교환하며 클라이언트 네비게이션창에서 TIE 시스템에 참가한 다른 아바타들과 서로 대화하고 춤을 추며 이동하게 된다. 그림 8은 실제 클라이언트의 네비게이션 창이다.

사용자는 서버에 접속하여 로그인을 할 때 실영상 아바타 또는 CG 아바타 형태로 가상공간에 참여하게 된다. 각각의 아바타는 키보드와 마우스를 사용하여 가상공간을 돌아다니게 된다. CG 아바타의 움직임은 키보드 입력으로 행해지며 뷰어의 이동은 키보드의 방향키로 움직인다.

그림 8 클라이언트의 네비게이션 창

4.3.1 실영상 기반 클라이언트 처리

각 클라이언트는 카메라로부터 캡처된 스트림을 H.263으로 실시간 인코딩하여 자신의 실영상을 서버로 전송한다. 서버에서는 QoS에 따라 트랜스코딩 된 스트림을 클라이언트에게 멀티캐스트를 통해 전송한다. 이때 실영상 기반의 클라이언트는 사용자 영상의 배경을 제거 한 후 빌보드 특성을 주어 가상현실내에 삽입하며, 3차원 영상을 통한 임의시점 깊이맵을 이용하여 임의시점 영상을 재구성한다.

아바타를 이용한 사용자는 아바타의 위치정보와 컨트롤 정보만을 전송한다. 컨트롤 정보는 아바타의 움직임을 정의한다. 실영상은 배경부분이 투명하게 처리된 후에 가상현실에 첨가되므로 크로마-키를 기법을 이용하여 배경을 제거 한 후에 서버로 전송하게 된다. 이 때 동영상은 투명도, 즉 알파채널을 지원하지 않기 때문에 배경부분을 특정색깔로 덮어씌운 후 전송하게 되며, 수신 단에서는 그 부분의 투명도를 1로 설정하는 방법으로 투명한 배경을 처리한다. 실영상은 JMF의 RTP 프로토콜을 이용하여 전송한다.

배경의 제거는 크로마-키를 이용하여 구현하였다. 다만, 동영상의 배경이 다른 이미지로 채워지는 것이 아니라 투명하게 설정되어 있어야 하므로 각 프레임별 이미지를 BufferedImage Object로 저장한 후에 배경부분의 투명도를 1로 설정하여 3D 오브젝트에 텍스처 매핑을 통해 투명한 배경의 동영상을 추출하였다. 그림 9는 Java3D와 JMF로 구현한 크로마-키 처리 과정을 나타내었다.

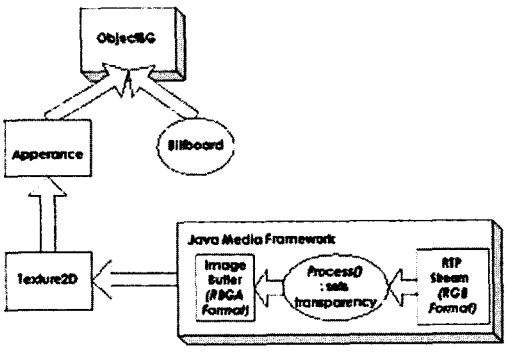


그림 9 Java3D와 JMF를 이용한 실영상 편집

4.3.2 CG 아바타

위에서 기술한 바와 같이 실영상은 3D Box에 택스처 매핑 되어 오브젝트의 형태로 가상현실에 삽입된다. CG 아바타는 이미 여러 형태로 디자인 되었으나 Java3D에

서 사용하기에 적합하도록 재구성하였다. 널리 사용되고 있는 3D 저작도구인 3D MAX studio를 사용하여 3D 아바타를 만든 후 Java3D에서 인식할 수 있는 obj파일(wavefront사의 파일포맷) 포맷으로 변환하여 사용하였다. Java3D로 구성된 아바타는 각 부분을 Branch-Group으로 정의하여 신체를 구성한다. 각 신체는 머리, 몸통, 팔, 팔뚝, 허벅지, 종아리, 발로 구분하여 구성되며 각각은 BranchGroup의 계층으로 조합된다. 아바타는 인체구조와 같은 관절을 가지고 있으므로 자연스럽고 다양한 동작 연출이 가능하다.

4.4 액티브 라우터

적응적 네트워크 환경을 위하여 TIE 시스템에 액티브 라우터[16]를 적용하였다. 이 시스템에서 액티브 라우터의 적응은 다음과 같이 동작한다. 액티브 라우터는 비디오 스트림에 대한 모든 데이타를 획득하고 스트림의 프레임 레이트를 바꿀수 있게 필터를 구동한다. 또한 액티브 라우터는 TIE 서버로부터 QoS 매트릭스를 받고 멀티캐스팅 모듈로부터 이웃 링크간의 네트워크 토폴로지 정보를 얻는다. 이 QoS 매트릭스와토폴로지 정보를 사용하여 멀티캐스팅을 하기위하여 비디오 스트림은 필터링된다. 네트워크상의 각 액티브 라우터는 정의된 방법에 따라 QoS 매트릭스를 사용하여 실시간으로 프레임 레이트를 변경한다. 만약 액티브 라우터에 여러 개의 링크가 있다면 멀티캐스트 그룹에 따라 스트림의 데이타 소스를 복사하여 전송한다. 그림 10은 액티브 라우터의 구성이다. 멀티캐스트 매니저는 지속적인 토폴로지 정보를 통해 자식노드의 멀티캐스트 그룹을 선택한다. 액티브 라우터는 네트워크 대역폭의 정보를 가지고 있어 이 데이타는 스트림 핸들러에 의해 비디오 스트림을 트랜스코딩한다. QoS 매니저는 서버로부터 QoS 매트릭스를 관리하고 이것을 다른 액티브 라우터에 브로드캐스트 한다. 네트워크상의 모든 액티브 라우터들은 이 QoS 매트릭스를 통해 모든 클라이언트들의 QoS 단계의 정보를 가지고 있다. 스트림 매니저는 QoS 메트릭스와 토폴로지 정보를 사용하여 스트림을 적절하게 트랜스코딩한다.

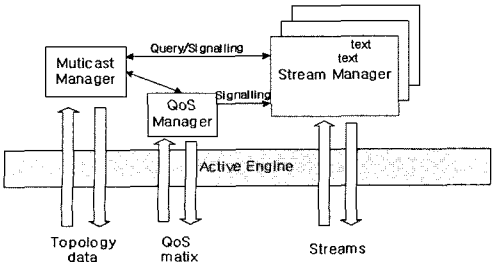


그림 10 액티브 라우터의 구성

5. 성능 분석을 위한 실험

5.1 실험배경

본 실험에서는 가상 댄싱 공간에 참여하는 다수의 클라이언트들의 QoS 단계를 각기 달리하여 서버, 라우터, 및 클라이언트에 부과되는 네트워크 트래픽에 대한 실험을 하였다. 서버에서는 QoS 매핑 곡선에서 결정된 아바타의 QoS 단계에 따라 스트림인 경우 프레임 레이트를 달리하여 트랜스코딩 하였다. 본 실험에 사용한 각 변수의 기본값으로 가상 댄싱 공간에 실영상 아바타로 참여하는 클라이언트의 수는 2명이며 CG 아바타로 참여하는 클라이언트의 수는 5명이다. QoS 단계는 각기 15fps(상), 10fps(중), 5fps(하)로 하였다. 또한 QoS를 적용했을 경우와 적용하지 않았을 경우의 실험을 통하여 계층적 QoS가 미치는 영향을 알아보았다.

5.2 실험 방법

그림 10에서 보듯이 TIE 시스템은 서버-라우터-클라이언트 구조로 실험을 하였다. 우선 CS(Configuration Server)는 Overlay 네트워크 실험환경을 위한 설정 환경 서버이다. 서버는 액티브 라우터(AR1)에 접속되어 각각의 클라이언트 접속을 기다린다. 각각의 클라이언트는 실영상 또는 CG 아바타로 서버에 접속하여 TIE 시스템에 참가한 다른 사용자들과 아바타로서 상호작용을 하게 된다. 실영상 아바타의 경우 본인의 실영상을 서버로 실시간 전송하게 되면 서버는 각각의 클라이언트의 뷰 포지션과 아바타들의 위치를 파악하여 QoS 단계를 계산하고 $m \times n$ 매트릭스에 저장하여 액티브 라우터로 보낸다. 이때, 서버로 들어오는 클라이언트들의 실영상 스트림을 액티브 라우터(AR1)에 전송하게 된다. 액티브 라우터는 서버로부터 실영상 스트림과 QoS 매트릭스를 받아서 각각의 클라이언트들에게 알맞은 스트림을 트랜스코딩하여 멀티캐스트하게 된다.

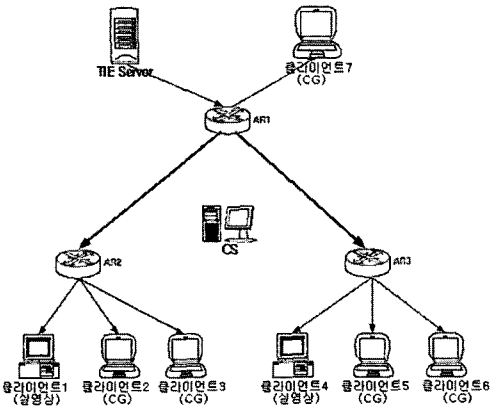


그림 11 네트워크 토폴로지

멀티캐스트는 233.0.0.x의 주소범위를 사용하였다. 우선 비디오 아바타가 선택이 되면 *AddressManager*로부터 233.0.0.x의 멀티캐스트 채널을 할당 받고, 이 채널을 통해 트랜스코딩된 비디오 아바타의 스트림이 멀티캐스트 된다. 앞에서 언급했듯이 멀티캐스트 채널의 관리는 비워져 있는 채널의 빠른 검색을 위하여 B+트리로 구현되어 있다.

본 실험은 각기 다른 QoS 단계로 클라이언트들의 위치를 조정한 후 경우의 수를 두어 총 3번의 시도를 통해 진행되었다. 첫 번째 시도는 클라이언트1(실영상)과 클라이언트2(실영상)의 뷰 포지션과 해당 실영상 아바타들의 위치를 조절하여 클라이언트1 에서는 클라이언트1의 실영상 아바타의 QoS를 '상'의 품질(15fps), 클라이언트2의 실영상 아바타의 QoS를 '중'의 품질(10fps)로 받게 하고, 클라이언트2에서는 클라이언트1의 실영상 아바타의 QoS를 '중'의 품질(10fps), 클라이언트2의 실영상 아바타의 QoS를 '중'의 품질(10fps)를 갖도록 하여 측정하였다. 두 번째 시도와 세 번째 시도 또한 비슷한 방식으로 각 실영상 아바타들의 QoS 단계를 달리하여 진행하였다.

5.3 실험 분석

액티브 라우터상에서 캡쳐된 패킷을 분석하면 실영상의 스트림이 15fps의 경우 약 89Kbps정도의 수치를 나타내고 있으며, 10fps의 경우 57Kbps, 5fps의 경우 약 29Kbps의 데이타가 수신된다. 즉, QoS 단계에 따라 액티브 라우터에서의 네트워크 부하는 현저하게 나타난다.

그림 12는 QoS를 적용한 경우(각기 다른 QoS 단계)와 적용하지 않은 경우 액티브 라우터에서의 평균 네트워크 부하를 측정 비교한 것이다. 그림 12에 나타나듯이 QoS를 적용한 경우는 적용하지 않은 경우보다 네트워크 부하가 상대적으로 적다는 것을 볼 수 있다. 계층적 QoS 모델을 적용하지 않은 경우, 모두 15fps의 실영상을 보내기 때문에 첫 번째 QoS 적용 경우(중,하,중,하)와 비교하면 약 1/3정도 차이가 나게 된다.

그림 13과 14는 각각의 클라이언트가 받게 되는 스트림의 수신률의 평균을 나타낸 것이다. 이 경우도 마찬가지로 QoS를 적용했을 때와 적용하지 않았을 때로 구분지었다. QoS를 적용한 경우는 적용하지 않은 경우보다 최대 1/3가량 줄어든 것을 볼 수 있다.

트랜스코딩을 액티브 라우터에서 하는 경우 QoS 단계에 맞게 트랜스코딩이 된다. 하지만 트랜스코딩을 클라이언트에서 할 경우 실영상의 원래 스트림을 받아야 하기 때문에 QoS를 적용하지 않은 것과 동일한 결과를 낳게 된다. 다시 말해서, 클라이언트에서 트랜스코딩을 하게 되는 경우, 15fps(약 90Kbps)의 데이타를 받아서 이를 다시 인코딩하기 때문에 만약 이 실영상을 5fps로

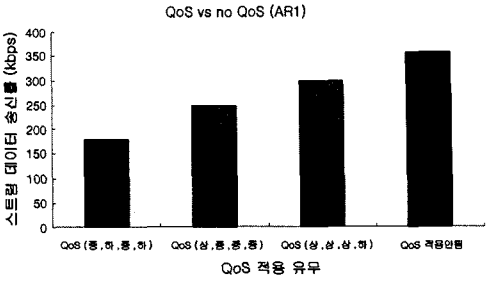


그림 12 액티브 라우터에서의 네트워크 부하

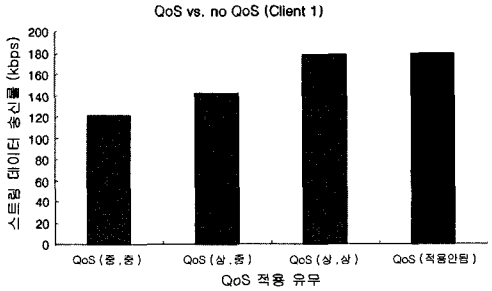


그림 13 클라이언트 1에서의 네트워크 부하

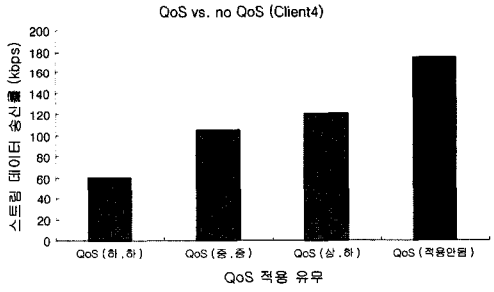


그림 14 클라이언트 4에서의 네트워크 부하

트랜스코딩 한다면 트래픽의 양은 90Kbps를 차지할 것이고 또한 트랜스코딩하는 부하까지 감안을 해야 한다. 이미 클라이언트는 가상환경, CG 아바타, 비디오 스트림등을 모두 렌더링 해야 하기 때문에 여기에 더 부하를 준다는 것은 성능저하를 가져올 수 있다. 하나의 클라이언트가 단지 한 개만의 스트림을 받는 것이 아니라 참가하는 모든 비디오 아바타의 실영상을 나타내야 하기 때문에 모든 스트림을 트랜스코딩하지 않은 상태에서 받는다는 것은 네트워크 트래픽의 양을 많게 하는 요인이 되며 QoS가 적용되지 않은 상태와 같다.

6. 시뮬레이션

본 시뮬레이션의 목적은 다수의 아바타가 참여한 가상 댄싱 공간에서 계층적 QoS가 미치는 영향을 살펴보고 더 나아가 계층적 QoS를 효율적으로 지원하기 위한 방

법을 모색하는 것에 있다. 본 시뮬레이션 환경은 Ultra-Sparc 60 450MHz에서 gcc를 사용하여 구현하였다.

가상 댄싱 공간에 참여하는 아바타의 초기 위치는 랜덤하게 결정된다. 아바타의 움직임은 사람의 걸음을 모델링 하였다. 자세히 설명하면 아바타는 현재 위치에서 8방향 중 임의의 방향으로 1스텝씩 움직이며 아바타의 위치는 1초마다 업데이트 된다. 본 시뮬레이션에서는 가상 댄싱 공간의 댄싱 플로워의 면적을 100 × 100 으로 설정하였고 각 아바타마다 좌표값(x_{avatar}, y_{avatar})을 갖게 된다. 아바타의 이동 속도율 a는 매 초마다 {0,1,2,3}중 임의의 하나를 선택하여 그 속도로 무작위로 움직이게 된다.

본 시뮬레이션에서는 QoS 매핑곡선의 상수값을 k=100, c=3으로 설정하였다. 본 시뮬레이션에서는 가상 댄싱 공간에 참여하는 클라이언트의 수, 경과 시간, QoS 단계의 개수, 가상 공간 내의 아바타의 밀집도 등을 변수로 두고 실험하였다. 여기서 QoS 단계 개수는 거리에 따라 관찰자가 보는 아바타의 화질을 몇 단계로 세분화 할 것인지 나타낸다. 서버에서는 QoS 매핑 곡선에서 결정된 아바타의 QoS 단계에 따라 $m \times n$ QoS 벡터를 생성하고 이 값에 따라 프레임레이트를 다르게 하여 트랜스코딩한다.

시뮬레이션을 살펴보면, 우선 댄싱 플로워 에서 하나의 아바타가 이동할 수 있는 최대 거리를 설정한다 ($D_{\max} = \sqrt{x^2 + y^2} = 141.42m$). QoS 매핑곡선에 해당하는 최대값과 최소값을 QoS 단계의 개수로 나누어 아바타간의 거리를 따져 QoS값을 설정한다. 시뮬레이션에 사용한 각 변수값으로는 아바타의 수는 1~50명, 경과시간은 100초, QoS 단계의 개수는 1~50이다. 앞에서 실험한 데이타를 바탕으로 H.263의 경우 1fps당 6Kbps의 데이타를 수신한다고 가정하고 최상의 품질인 경우 15fps를 받는다고 가정하여 약 90Kbps의 데이타를 수신하게 된다.

본 논문의 시뮬레이션에서는 클라이언트의 렌더링 오버헤드나 멀티캐스트 그룹 변경으로 인한 오버헤드는 고려하지 않는다.

그림 15는 클라이언트가 증가하였을 때 네트워크 트래픽을 QoS 단계의 개수별로 나타낸 것이다. 그림 15를 살펴보면 클라이언트가 증가할수록 계층적 QoS 모델을 적용한 것과 적용하지 않은 것의 차이는 매우 크다는 것을 알 수 있다. 특히, QoS 단계의 개수를 50개로 나눈 경우 QoS가 없는 것과 비교할 때 약1/3가량 줄일 수 있으므로 다수의 사용자가 참여하는 가상 댄싱 공간을 효과적으로 지원할 수 있다는 것을 보여준다. 단, QoS 단계의 수를 늘릴수록 패킷의 개수는 더 감소하였으나 빈번한 멀티캐스트 그룹 변경으로 인한 오버헤드를 고려한다면 무조건 QoS 단계를 세분화하는 것보다는 현재 가상 댄싱공간에 적합한 QoS 단계로 세분화하는 것이 중요하다.

그림 16, 17, 18, 19는 QoS 단계에 따른 데이타의 수신량을 클라이언트의 수 별로 나타낸 것이다. 그림 16은 클라이언트의 수가 10일 때 각각의 QoS단계의 개수별 데이타 수신량이다. 여기서 QoS의 단계의 개수를 1~5, 6~10, 11~15, 16~20, 21~25, 26~30의 총 6개 구간으로 묶어서 각각을 평균 낸 수치이고 붉은색으로 표시된 것이 최소값이다. 그림 17, 18, 19도 각각 클라이언트의 수가 20, 30, 40일 때 각각의 QoS 단계 개수별 데이타 수신량을 나타내고 있다.

그림 16의 경우 QoS의 단계 개수가 6~10사이의 수신 데이타 평균은 약 5.1Mbps이고 다른 QoS의 단계 개수 구간들과 비교했을 때 최저값을 나타낸다. 따라서 클라이언트의 수가 10일 경우 QoS 단계의 수를 6~10 사이로 세분화해 주는 것이 효율적이라는 것을 알 수 있다. 그림 17의 경우 클라이언트의 수가 20일 때 QoS의 단계를 16~20사이에서 나누어 주는 것이 가장 효율적이라는 것을 알 수 있다. 하지만 이 구간의 양 옆인 11~15 및 21~25 구간의 경우 상대적으로 적은 수치의 데이타를 수신한다는 것을 알 수 있다. 그림 18의 경우 클라이언트의 수가 30일 때 QoS의 단계를 21~25사

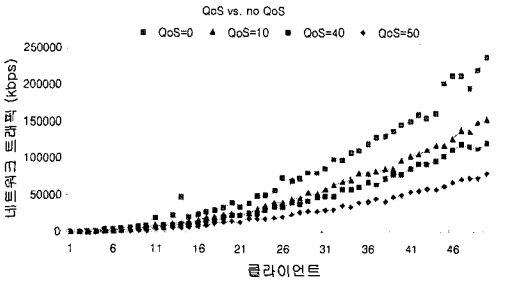


그림 15 클라이언트 수의 변화에 따른 데이타량

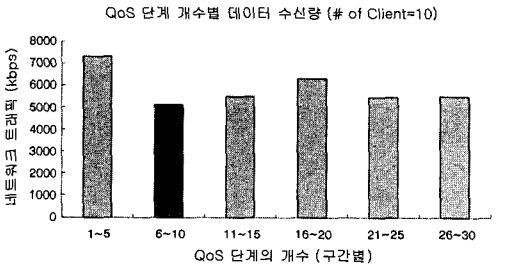


그림 16 QoS 단계 개수별 데이타 량
(# of Client=10)

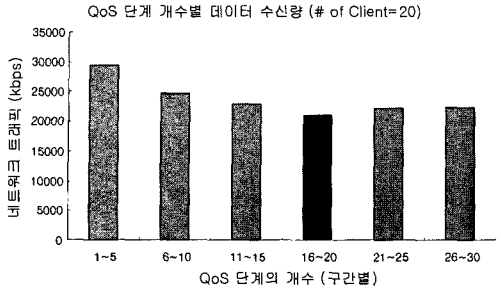


그림 17 QoS 단계 개수별 데이타 량
(# of Client=20)

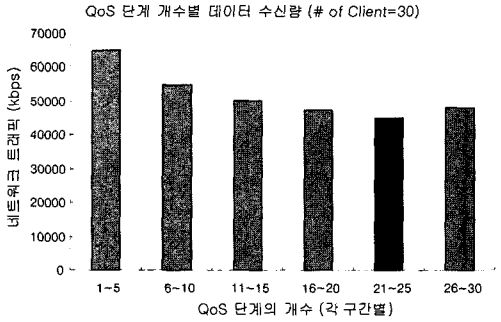


그림 18 QoS 단계 개수별 데이타 량
(# of Client=30)

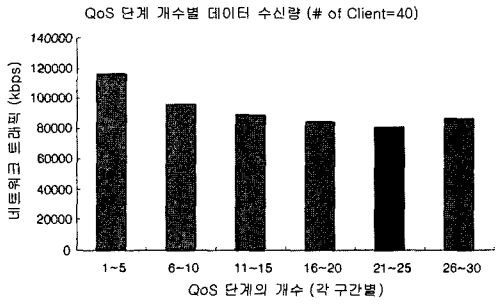


그림 19 QoS 단계 개수별 데이타 량
(# of Client=40)

이에서 나누어 주는 것이 바람직 하다. 또한 그림 19의 경우도 클라이언트의 수가 40일 때 QoS의단계를 21~25에서 세분화하는 것이 효율적이라는 것을 알 수 있다. 위의 4가지 경우를 비교할 때, 클라이언트의 수가 증가할수록 QoS의 단계의 수도 증가하는 것이 효율적인 네트워크 자원의 활용이라는 것을 알 수 있다.

그림 20은 가상 댄싱 공간에서 30명의 클라이언트가 참여하였을 때 아바타의 밀집도(%)에 따른 패킷 전송량 변화를 측정한 것이다. 아바타의 밀집도(%)란 전체 아바타의 개수에서 선택된 구역에 속한 아바타의 개수의

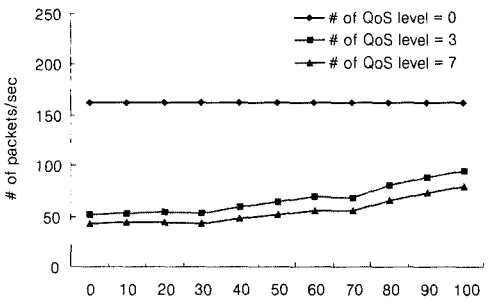


그림 20 아바타의 밀집도에 따른 패킷 전송량 변화
(# of clients = 30)

비율을 나타낸다. 가상 댄싱 공간을 동일한 크기를 갖는 9개의 구역으로 분할하고 한 구역을 선택하여 아바타의 밀집도에 따라 아바타를 할당하고 나머지 아바타는 선택된 구역을 제외한 나머지 구역에 무작위로 할당한다. 따라서 특정 지역에 속한 아바타의 밀집도를 점차적으로 높이면서 가상 댄싱 공간에서 발생하는 패킷의 개수를 측정하였다.

그림 20과 같이 계층적 QoS를 적용하지 않은 경우, 아바타의 밀집도에 관계없이 네트워크를 통해 전송되는 패킷의 개수가 동일하다. 이에 반해 계층적 QoS모델을 적용한 경우는 뷰의 허용 범위를 벗어난 아바타의 스트림은 받지 않으므로 가상 댄싱 공간에서 전송되는 패킷의 개수를 줄일 수 있을 뿐만 아니라 특정 구역에 아바타가 밀집해 있더라도 관찰자의 화질에 따라 아바타의 화질이 다르게 결정되므로 네트워크를 통해 전송되는 패킷의 개수를 효과적으로 줄일 수 있다. 또한 아바타의 밀집도가 증가해도 패킷의 개수가 완만하게 증가하므로 아바타의 밀집도가 높은 가상 댄싱 공간을 효과적으로 지원해 줄 수 있다.

7. 결 론

계층적 QoS 모델은 관찰자의 시점에서 보여지는 아바타의 위치에 따라 사용자가 느끼는 화질을 가상 공간 내의 아바타의 화질로 결정하므로 결과적으로 사용자의 화질을 손상하지 않으면서 네트워크를 통해 전송되는 패킷의 개수를 줄일 수 있었다. 또한 QoS 단계 개수를 세분화한 경우에는 네트워크를 통해 전송되는 데이타량을 좀 더 감소시킬 수 있었으나 멀티캐스트 그룹의 빈번한 변경으로 인한 오버헤드를 고려해야 하므로 구현한 가상 댄싱 공간에 적합하게 QoS 단계를 나누어야 한다

계층적 QoS 모델은 다음과 같은 특징을 가진다. 첫째 사용자가 느끼는 화질에 따라 아바타를 렌더링 하므로 클라이언트가 처리할 스트림을 감소시킬 수 있다. 특히

실영상과 3D 오브젝트가 참여하는 가상 공간에서는 렌더링과 스트림 처리에 대한 부하가 리소스의 대부분을 차지하므로 클라이언트의 프로세싱 로드를 감소시키는 것이 중요하다. 두 번째 관찰자로부터 거리에 따라 최소한의 QoS가 보장되는 스트림만 받기 때문에 필요 없는 스트림 처리로 인한 로드를 줄일 수 있다. 셋째 아바타의 영향 범위와 아바타간의 거리에 의해서 볼륨의 크기가 조절되므로 아바타의 이동에 의해 소리의 변화를 느낄 수 있어 가상 공간내의 사용자의 몰입감을 증가시킨다.

본 논문은 가상 댄싱 공간에서 계층적 QoS를 적용하여 네트워크를 통해 전송되는 패킷의 개수를 감소시켰다. 특히 다수의 사용자가 참여하고 아바타의 밀집도가 높은 가상 댄싱 공간에서 좋은 성능을 보여주었다.

본 논문의 의의를 다음과 같이 요약할 수 있다. 먼저 배경이 제거된 실영상을 가상 공간에 참여 시킴으로 사용자의 몰입감을 증대시켰다. 가상 공간 내에 실영상을 참여시킴으로서 사용자는 아바타 체험에 그치지 않고 현장에서 교육하는 것과 같은 효과를 줄 수 있다. 즉 아바타가 아닌 자신의 분신이 가상 공간에 참여하여 다양한 활동을 할 수 있는 가능성을 열어 놓았다. 또한 계층적 QoS를 적용하여 사용자가 많을 때 네트워크상에 부가되는 부하를 효율적으로 줄일 수 있다.

참 고 문 헌

[1] C. Carlsson and O. Hagsand, Dive-a multi-user virtual reality system. In Virtual Reality Annual International Symposium, IEEE, Sept. 1993, pages 394-400.

[2] O. Hagsand. Interactive Multiuser VEs in the DIVE system, In IEEE Multimedia, Spring 1996, pages 30-39.

[3] E. Frecon and M. Stenius. Dive: A scalable network architecture for distributed virtual environments. Distributed Systems Engineering Journal, Sept. 1998, 5(3):91-199.

[4] A. Steed, J. Mortensen, and E. Frecon. Spelunking : Experiences using the dive systems on cave-like platforms, Immersive Projection Technologies and Virtual Environments 2001, Springer-Verlag, May 2001, pages 153-164.

[5] D. R. Pratt, J. C. Walter, P. T. Warren, M. J. Zyda. NPSNET : JANUS-3D providing three-dimensional displays for a two-dimensional combat model, AI, Simulation, and Planning in High Autonomy Systems, 1993. 'Integrating Virtual Reality and Model-Based Environments,' 4th Annual Conference, Sept. 1993, Pages 31-37.

[6] M. Capps, D. McGregor, D. Brutzman, M. Zyda. NPSNET-V. A new beginning for dynamically extensible virtual environments, Computer Graphics and Applications, IEEE, Volume:20, Issue:5, Sept. 2000, Pages:12-15.

[7] M. R. Macedonia, M. J. Zyda, D. R. Pratt, D. P. Brutzman, P. T. Barham. Exploiting Reality with Multicast Groups: A Network Architecture for Large Scale Virtual Environments, 1995 IEEE Virtual Reality Annual Symposium, Pages:2-10, 1995.

[8] C. Greenhalgh, S. Benford. MASSIVE: a Collaborative Virtual Environment for Tele-conferencing, ACM Transactions on Computer-Human Interaction. Sept. 1995, Volume 2 Issue 3: Pages: 239-261.

[9] C. Greenhalgh, G. Reynard. A QoS Architecture for Collaborative Virtual Environments, Proceedings of the seventh ACM International Conference on Multimedia, 1999: Pages : 121-130.

[10] S. Benford, C. Greenhalgh, D. Snowdon, A. Bullock. A Poetry Performance in a Collaborative Virtual Environment, IEEE Computer Graphics and Applications, May 2000: Pages: 66-75.

[11] S. Benford, C. Greenhalgh, G. Reynard, C. Brown, and B. Koleva. Understanding and constructing shared spaces with mixed-reality boundaries, ACM Transactions on Computer-Human Interaction, 5(3):182-223, Sept. 1998.

[12] J. Pubrick C.G., Extending Locales: Awareness Management in MASSIVE-3. URL:www.crg.cs.nott.ac.uk/research/systems/MASSIVE3, September 1999.

[13] J.C.S. Lui, M.F. Chan. An efficient partitioning algorithm for distributed virtual environment systems, Parallel and Distributed Systems, IEEE Transactions, Volume:13 Issue:3, Mar. 2002, Page(s): 193-211.

[14] Shun Yan Cheung, M.H. Ammar, Xue Li. On the use of destination set grouping to improve fairness in multicast video distribution, INFOCOM 2000. 19th Annual Joint Conference of the IEEE Computer and Communications Societies. IEEE, Volume: 1, 26-30 Mar. 2000.

[15] Sandeep Singhal M. Z. Networked Virtual Environments : Design and Implementation(1999): Addison-Wesley/ACM Press.

[16] D. Tennenhouse. A Survey of Active Network Research, IEEE Communications, Jan. 1997.

김 진 용

2001년 한양대학교 전자전자통신전파공학과 학사. 2002년~현재 한양대학교 전자통신전파공학과 석사과정. 관심분야는 멀티미디어 시스템, QoS, 스트리밍, 분산 가상환경

원 유 집

1990년 서울대학교 자연과학대학 계산통계학과 학사. 1992년 서울대학교 자연과학대학 계산통계학과 전산학 석사. 1997년 University of Minnesota 전산학 박사. 1997년~1999년 Server Performance Analyst, Intel Corp. 1999년 3월~현재 한양대학교 공과대학 전자전기컴퓨터 공학부 교수. 관심분야는 멀티미디어 시스템, 멀티미디어 네트웍, 인터넷 프로토콜, 성능평가이론, 데이타베이스, 운영체제

김 범 은

2000년 한양대학교 전자전자통신전파공학과 학사. 2002년 한양대학교 전자통신전파공학과 석사. 2002년~현재 삼성전자 DM총괄. 관심분야는 멀티미디어 시스템, QoS, 운영체제, 분산가상환경, 이종환경

박 종 일

1987년 서울대 전자공학과 졸업. 1989년 서울대 전자공학과 석사. 1995년 서울대 전자공학과 박사. 1995년~1996년 한국방송개발원 선임연구원. 1996년~1999년 일본 ATR지능영상통신연구소 연구원 1999년~현재 한양대학교 전자전기컴퓨터공학부 교수. 관심분야는 컴퓨터비전, 컴퓨터그래픽스, 가상현실

박 용 진

1969년 와세다대학교 전자통신공학과 1972년 와세다대학교 공학석사. 1978년 와세다대학교 공학박사. 1979년 현재 한양대 전기전자컴퓨터공학부 교수. 1983년 Univ. of Illinois, Urbana 전산학과 방문교수. 1991년 영국 Kent 대학 방문교수. 현재 본학회 수석부회장, IEEE Region 10 총무이사 관심분야는 컴퓨터통신