

감성 공학 환경을 지원하는 soft-real time 통신 프로토콜의 설계

진종욱, 고희동
한국과학기술연구원 영상미디어 연구센터

A design of soft-real time communication protocol for the environment of gamsung engineering

Jongwook Jin, Heedong Ko
KIST Image media research center

요약

다중 감각 실험 환경을 구축시에 시청각 뿐 아니라 열감각, 촉각, 후각등 다중 감각등의 제시가 필요하다. 이러한 실험 환경은 작업의 부하를 분산하고, 모듈 개발 방법을 도입하기 위해서 분산 시스템으로 구축되는데 이에 추가적으로 실시간 동작 속성을 가지는 프로토콜이 필요하다. 본 연구는 이러한 실시간 동작 속성을 가지며 개별 모듈의 용이한 개발의 지원과 고장시의 대책으로서 임의의 모듈의 고장시의 대처 방안과 제어 모듈의 프로그램 용이성을 제공하는 프로토콜을 설계하였다.

다양한 시뮬레이션 재구성 능력을 제공할 수 있는 펄드의 동적 바인딩을 지원하며, 동일 모듈의 연결로 새로운 모듈을 합성할 수 있는 능력을 가질 수 있는 분산 실험 시스템을 설계하였다.

1. 서론

감성 공학 차원에서 실험 제시 시스템은 다중 감각 제시 모듈의 조합으로 피험자에게 제시되어야 하며 따라서 시청각 위주의 VR 시스템보다 다양한 제시 모듈이 필요하다.

모듈을 연결하는 기본 기능 이외에도 감각 제시기의 특성을 고려한 시스템 통합 방법의 제시가 필요하다. 운영 체제관점에서의 감성 공학 모듈들의 특성은 운용 시간에 따라서 크게 실시간 응용이 가능한 것과 실시간 응용이 불가능한 것으로 두개로 나눌 수 있으며,

실시간 응용이 가능한 것도 매 시물레이션 클릭에서 수행되는 것과 시물레이션 전체의 정보 처리로 나눌수 있다.

Classes	Major process
	Type of communication
Short term	Update operation, response for input
	Online/Network/Tight couple system
Mid term	Data processing simulation database, control simulation
	Online/Network/File System/Loose couple system
Long term	Build up virtual environment, analysis of simulation data
	Offline/File System

표 1 운영에 따른 실험 모듈의 분류 및 연결의 유형

2. Support real time constraints of gamsung engineering

다중 감각 제시의 실시간 제시를 하기 위해서는 감각 제시 장비의 특성에 맞는 통신 모듈이 필요하다. 업데이트 레이트는 감각마다 다양하다. 시각은 최소 15Hz (스테레오 지원시 최대 120hz업데이트)가 필요하며 열환경 제시 장비중의 일부로 공기제어 장비는 물리적 한계로 업데이트 레이트가 1Hz미만으로 응답 속도가 느린편이다. 반면에 힘궤환 장비는 보다 높은 주기로 업데이트 (수 khz)해야 현실감이 있으며, 이와 같은 높은 업데이트 레이트의 처리는 분리되어 있는 처리기와 독립된 고속의 버스가 필요하다. 본 연구에서는 S/W적으로 처리할 수 있는 업데이트를 한계로 한다

대부분의 컴퓨터 통신이 데이터의 신뢰성을 보장하기 위해서 에러 복구를 수행하기 때문에 응답 속도면에서 희생이 불가피하다. 그러나 감각 제시 시스템에서는 응답 시간의 통신에 의한 오버헤드를 최소화하는 것이 중요하다.

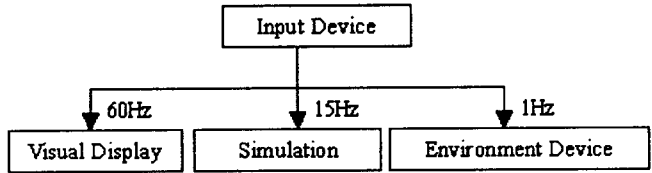


그림1 모듈의 연동 및 시간 제약 사항의 다양성

데이터를 반영하는 방법도 감각 업데이트 레이트에 따라서 다양하다. 업데이트 레이트가 낮은 장비의 경우에는 예측 시스템이 필요하다. 데이터의 속도와 가속도를 분석하면서 미리 예측하면 감각 제시에서 좋은 결과를 얻을 수 있다. 이와 반면에 시각이나 청각 제시의 경우 예측 데이터로 제시하지 않아도 높은 응답 속도를 얻을 수 있다.

2.1 Real time distribution of source data

실시간 요구 사항을 지키기 위한 전달 방법은 크게 두가지로 나누어 진다. 하나는 받는 측 시작 통신 방법 (Receiver initiate)으로 모듈이 필요한 필드를 원하는 주기로 요청하여 소스 모듈이 이에 대해 응답하는 방법과 이에 반대로 보내는 측 시작 통신 방법 (Sender initiate)으로 소스 모듈이 필요한 주기로 계속적으로 전달하는 방법이 있다. 전자는 응답 속도에 요청하는 시간이 포함되기 때문에 시각 제시기와 같은 높은 주기를 가져야 하는 모듈에는 통신 시간 부담이 커진다. 따라서 우리는 후자의 모델을 선택하였다.

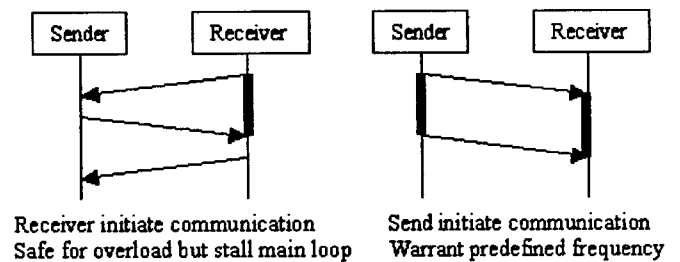


그림 2 대표적인 통신 모드

2.2 Bypass I/O invocation in simulation loop

설계된 모델은 통신 Agent를 두어서 통신을 전달하기 때문에 시뮬레이션 루프는 상대적으로 부담이 적은 공유 메모리 접근 방법으로 필드를 접근할 수 있다. Agent는 Simulation Loop이 I/O에 종속되지 않고 최신의 데이터를 서비스 해준다. Sender도 Receiver와 같은 구조로 구성되어 있다.

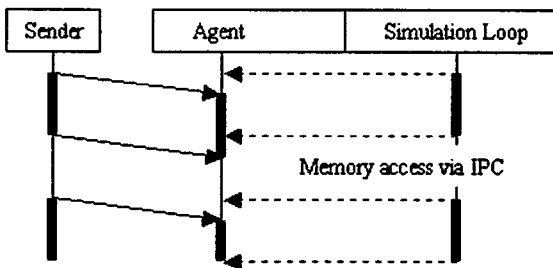


그림 3 Communication agent (Receiver side)

2.3 Multiple update rate support

보통의 시뮬레이션은 한 모듈이 모듈의 집합을 처리하기 때문에 모듈의 분산이 불가피해질 때, 주 프로세싱 시간에 종속적이면 융통성있게 다중 업데이트 레이트를 맞추어 줄 수 없다. 따라서 주 프로세싱과 분리된 프로세스나 스레드가 필요하다.

이 문제를 해결하는 방법은 모든 모듈에 대응되는 하나의 프로세스/스레드를 만드는 것이다. 프로세스를 많이 만들면 부담이 크고, 스레드의 경우에는 스레드의 스케줄링의 부정확성을 고려해보면, 하나의 에이전트가 운영성에서 유리하다고 볼 수 있다.

본 연구에서는 하나의 에이전트를 구성하여 루프 안에서 모듈이 필요로하는 업데이트 레이트의 최대 공약수 시간 만큼 기다리고 필요한 모듈에게 필드를 전달한다.

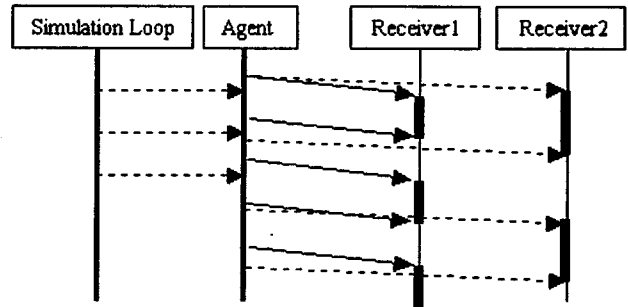


그림 4 Warrant the multiple update rates (Sender side)

```

While (1)
{
    Sleep (업데이트 레이트의 최대 공약수)
    While (logged module loop)
    {
        While (registered field loop)
        {
            If (TimeToSend (module, field))
                Sendifchanged field to module
        }
    }
}
    
```

알고리즘 1 Agent send algorithm

3. Dynamic field binding for easy to program and to organize system

모듈과 모듈이 필요로 하는 필드의 연관 관계가 시뮬레이션 실험 목표에 따라서 가변적으로 변화하므로 하나의 모듈이 다중 목표로 작성되는 경우가 많다. 예를 들면 입력장치의 해석이 각 실험마다 달라서 요구하는 정보의 내용이 다른 경우에는 입력장치의 모듈이 값을 전달하는 방법을 다르게 할 수 있는 메커니즘이 필요하다.

이러한 메커니즘을 지원하기 위해서 실험 환경과 동적으로 모듈의 운영 상황 정보를 전달해줄 분산 Coordinator가 필요하다. Coordinator의 역할은 모듈의 운영 상태와 위치 전달을 모듈에 전달해준다. 이 메커니즘에 의해서 임의의 모듈의 역할은 동적으로 재설정된다.

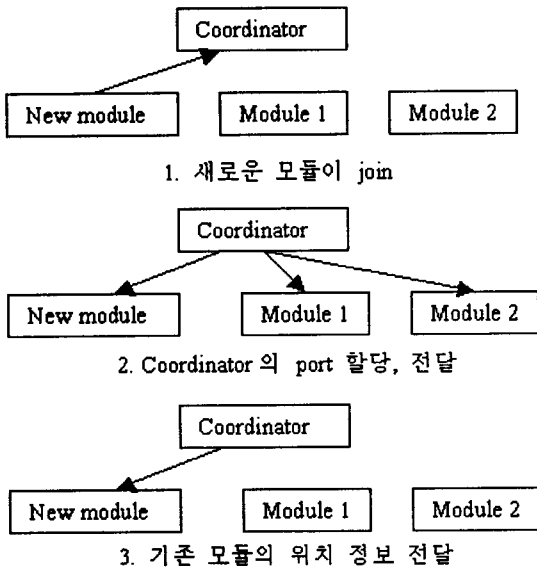


그림 5 새로운 모듈의 조인시의 Coordinator의 동작

새로운 모듈이 참여시의 Coordinator의 동작은 port 할당을 통해서 새로운 모듈의 위치를 다른 모듈과 충돌없이 할당하며 할당된 번호를 모듈명과 함께 다른 이미 참여되어 있는 모듈에게 전달하여 Agent들이 재설정할 수 있는 정보를 제공한다. 새로 참여된 모듈은 이미 참여하고 있는 모듈의 위치 정보를 Coordinator로부터 전달 받는다.

이와 같은 재 설정 방법으로 Receiver 모듈의 고장이나 종료 등으로 Coordinator가 Sender 모듈에 모듈의 운영상황을 전달하면 그 Receiver 모듈에 대한 전송을 중지한다. 다시 Receiver 모듈이 운영에 참가하게 되면 Sender 모듈은 다시 필드를 재전송하게 되는데 모듈 작성자가 제작하는 Sender모듈은 이러한 상황에 무관하고 Sender의 Agent가 이러한 복잡성을 해결한다.

4. Implementation Issue

4.1 Network transfer protocol

여러 개발 플랫폼에 공개되어 있는 TCP/IP를 이용했다. UDP 통신 프로토콜은 TCP 방법에 비해서 신뢰성은 떨어지지만 응답 속도가 빠르며, Connection 단계가 필요 없으므로 다중 Sender 모듈이 Receiver 모듈에 쉽게 접근하기 편하다. 코디네이터와 모듈간의 연결은 신뢰할 수 있는 TCP 프로토콜을 이용하였다.

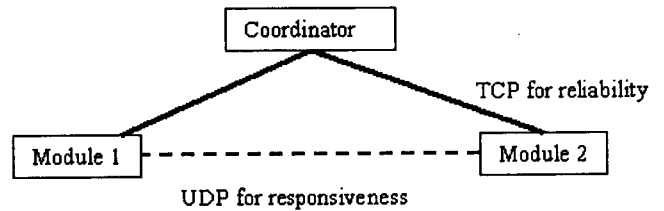


그림 6 Using TCP/IP transfer protocols

4.2 Combination of module

Coordinator의 도움으로 기존 모듈의 합성으로 새로운 모듈을 만들 수 있는데, 예를 들면 사용자의 몰입감을 증대시키기 위한 다중 채널 디스플레이를 단일 채널 비주얼 워크스테이션의 클러스터로 구축할 수 있다.

4.3 Synchronize in m-sec units

현재 운영체제에는 신뢰할 수 있는 수 m-sec단위의 정밀성을 가지는 타이머가 존재하지 않는다. 이유는 프로세스 개수가 동적으로 달라져서 프로세스 스위칭 시간과 I/O 시간이 동적으로 변화하기 때문에 m-sec단위의 타이머는 주기마다 가변적으로 변화할 수 있다. 따라서 Agent 코드안에서 이러한 시간 에러와 loop안에서 걸린 시간을 보정하는 부분이 있어야 한다. 모듈 Boot-up time이나 모듈 재설정 시에 클럭 조정의 단계가 필요하다.

4.4. Divide by 1

Agent 기법을 운용중에 모듈과 필드가 추가되어 업데이트 레이트의 최대 공약수가 1에 가깝게 작아지면 Sender agent가 계속적으로 루프를 반복하게 되며 프로세스나 스레드의 프로세서 점유율이 높아지고 또한 동기화 기법에 보장이 없고 동기화 호출을 상대적으로 많이 하기 때문에 시간적 오차가 커진다. 이를 피하는 강제적인 방법은 최대 공약수를 보장하는 더 작은 업데이트 레이트로 변환하는 것이다. 그러나 보정으로 피할 수 없는 경우에는 모듈과 필드가 새로이 추가될 때 제한적으로 프로세스 및 스레드를 복수개 생성하여 업데이트 모듈 집합을 파티션하는 것이 유리하다. 파티션의 고려점은 최대 공약수를 최대한으로 하는 적당한 셋으로 분할 하는 방법이 필요하다.

5. Conclusion and future works

종합적인 가상현실 실험 시스템의 구축을 위해서 기존의 컴퓨터 통신과 모니터링을 혼합한 agent를 개발하고 현재 입력 디바이스와 시각 및 열감각, 후각 표시 모듈에 적용하였고 개발되고 있는 시스템에 대한 확장도 가능한 통신 기반 구조를 구축하였다.

감성 실험에 참가하는 모듈들을 연동하기 위해서는 이러한 연결 프로토콜의 연구가 선행되어야 하며, 이러한 통신 프로토콜을 UNIX기반의 SGI Multi-processor workstation에서 적용한 결과, 통신에 의한 오버헤드를 많이 줄일 수 있었다.

Single-processor환경에서라도 매주기마다 프로세서가 idle한 시점을 이용하여 통신 프로세스되므로 프로그래머가 주 루프에서 시간제약을 지켜야 하는 복잡성을 agent가 대신하여 개발이 용이해지는 유리한 잇점을 가진다.

6. 참고 문헌

- [1] 박경동, 이상민, 교회동 “감성공학을 위한 실시간 워크쓰루 시스템 개발”, HCI'97 학술대회 pp. 388-395, 1997
- [2] 박창훈, 박경동, 교회동 “가상현실 시스템을 위한 외부 모듈 인터페이스의 개발” HCI'99 학술대회 pp. 351-355, 1999
- [3] Bricken, W. “VEOS: preliminary functional architecture” ACM Siggraph'91 Course Notes, Virtual Interface Technology, 46-53
- [4] M.Macedonia, D.Pratt and M.Zyda, “NPSNET: A network software architecture for large scale virtual environments” Presence. Vol.3 No.4. pp. 302-311, 1994
- [5] Stevens “Unix network programming” Prentice Hall
- [6] Quinn, Shute “Windows Sockets Network Programming” Addison Wesley