석 사 학 위 논 문

인터넷 기반 퍼스널 로봇 시스템의 제어 구조 설계

한 국 현(韓 國 鉉)
전기 및 전자공학과 (제어공학 전공)
한 국 과 학 기 술 원

1999

인터넷 기반 퍼스널 로봇 시스템의 제어 구조 설계

Control Architecture Design of Internet-Based Personal Robot Systems

Control Architecture Design of Internet-Based Personal Robot Systems

Advisor : Professor Jong-Hwan Kim

by

Kuk-Hyun Han

Department of Electrical Engineering (control program)
Korea Advanced Institute of Science and Technology

A thesis submitted to the faculty of the Korea Advanced Institute of Science and Technology in partial fulfillment of the requirements for the degree of Master of Engineering in the Department of Electrical Engineering (control program)

Taejon, Korea
1998. 12. 31.
Approved by

_____

Professor Jong-Hwan Kim
Major Advisor

인터넷 기반 퍼스널 로봇 시스템의 제어 구조 설계

한 국 현

위 논문은 한국과학기술원 석사 학위논문으로 학위논문심사위원회에서 심사 통과하였음.

1998년 12월 28일

심사위원장 김 종 환 (인)

심사위원 정 명 진 (인)

심사위원 임 종 태 (인)

사랑하는 가족에게 드립니다.

MEE 973724 한국현. Kuk-Hyun Han. Control Architecture Design of Internet-Based Personal Robot Systems. 인터넷 기반 퍼스널 로봇 시스템의 제어 구조 설계. Department of Electrical Engineering. 1999. 80p. Advisor Prof. Jong-Hwan Kim.

ABSTRACT

This thesis proposes a novel control architecture for internet-based personal robot systems. The personal robot considered in this thesis is a kind of service robots which can be used for a person's convenience in the house. It has a wireless LAN system for the internet remote control. Users can control the personal robot using a simulator located at local sites. Since the internet time delay is affected by the number of nodes and the internet loads, it is variable and unpredictable so that a large internet delay makes some control inputs distorted. In order to control the personal robot without the influence of the internet delay, a novel internet control architecture is designed. This architecture guarantees that the robot can avoid obstacles, and reduces the path error and the time difference between the virtual robot in a local site and the real robot in a remote site. It consists of a posture estimator based on the posture information from the remote site, a command filter constructed by a command queue and a command generator, a path generator for each sample time and a path-following controller to recover the time difference. The difference between a virtual environment in local sites and a real environment in remote sites can increase a path error between a virtual robot and a real robot. To solve this problem, a virtual environment supervisor with a sensor feedback from the real robot is proposed. The results of simulations and experiments demonstrate the effectiveness and applicability of the proposed internet control architecture.

목 차

1. 서 론

1.1 연구 동기

과거의 로봇 시스템은 산업 현장에서 용접, 페인팅, 운반, 부품의 조립 등과 같은 단순, 반복적인 일을 수행하는 분야에 주로 사용되었다. 하지만 로봇에 대한 관심의 증대와 많은 연구에 힘입어 인간의 단순한 노동력을 대신해주는 일 뿐만 아니라, 심해나 우주 탐사, 원격 수술과 같은 다양한 분야로 그 응용 분야가 확대되고 있으며, 최근에는 인간의 일상 생활에 편의를 제공하는 가정용 로봇이 등장하기까지에 이르렀다. 현재의 로봇 시스템의 발전 과정은 컴퓨터의 그것과 매우 흡사하다. 컴퓨터가 등장한 초기에는 큰 부피와 비싼 가격 때문에 일반인들이 사용하기에는 적합하지 못했다. 하지만 컴퓨터 관련 기술이 발달함에 따라 부피가 작아지고 값이 낮아지면서, 현재는 개인이 사용하기에 적합한 퍼스널 컴퓨터가 등장하였다. 복잡한 연산을 대신해주는 도구였던 컴퓨터는 점차 인간에게 편의를 제공해주는 도구로 발전하고 있으며, 컴퓨터의 발전과 함께 로봇의 발전이 병행되어 이루어지고 있다.

사람이 같은 공간에 있는 로봇만을 제어할 수 있었던 과거의 고정 관념은 원격 로보틱스(telerobotics) 분야의 등장으로 사라지게 되었다. 그 기술력은 지구에서 화성의 표면 위에 위치하고 있는 탐사 로봇[1]을 직접 제어하는 것을 가능하게 만들었다. 이와 같은 원격 제어 기술은 점차 보편화되어지고 있다. 미래에 원격 제어가 가장 필요한 분야 중의 하나는 퍼스널 로봇이다. 현재 모든 가정용 전자 제품들이 하나의 네트워크를 형성하며 발전해 나가고 있다. 이와 같은 추세는 가정의 네트워크를 관리할 수 있는 원격 제어가 가능한 퍼스널 로봇의 등장을 앞당길 것이다. 또한 주택을 벗어나 보내는 시간이 많은 현대에는 원격 제어 기능이 필수적이다. 원격 제어를 위한 수단은 인터넷망, 전화선망, 인공위성 등 여러 가지가 있다. 이와 같은 수단 중에서 보편화 가능성이 높고, 다양한 정보의 전송이 가능한 매체는 인터넷망이다. 인터넷망은 세계 어느 곳이나 연결되어 있으며, 저렴하게 이용할 수 있다. 또한 웹 브라우저를 이용함으로써 프로그램이나 사용

자 인터페이스 등의 표준화를 쉽게 이룰 수 있다.

본 논문에서는 인터넷 기반 퍼스널 로봇 시스템의 원격 제어에 적합한 제어 구조를 제안함으로써 가까운 미래에 실현될 퍼스널 로봇의 기반 기술을 이루는 것을 목표로 한다.

1.2 연구 동향

인터넷망이 보편화됨에 따라 90년대 중반에 이르러 인터넷과 로봇을 접목시키는 인터넷 로보틱스(internet robotics) 분야[6]가 등장하게 되었다. 세계 어느 곳이나 쉽고 빠르게 접근할 수 있다는 인터넷의 장점 때문에 인터넷 로보틱스 분야는 활발한 연구가 진행되었다. Taylor[6]는 인터넷을 통해 전송되는 화상을 보고 로봇 팔을 제어하여 블록을 쌓는 시스템을 구현하였고, Goldberg[4]는 공기 노즐과 비젼 시스템을 이용하여 모래 속에 숨겨진 물체를 찾아내는 원격 정원(TeleGarden) 시스템을 구현하였다. Chen[7,8]은 센서 기반 이동 로봇을 인터넷을 통해 제어할 수 있는 시스템을 구현하였고, 이하섭[13]은 웹 상에서의 원격 전시 시스템을 구현하였다. 하지만 구현된 대부분의 인터넷 기반 시스템은 상위 레벨 명령을 주면 로봇은 그 명령에 해당하는 작업을 스스로 수행하도록 구현되었다. 인터넷 시간 지연은 직접 제어 방식을 구현하는데 있어서 어려움을 주는 요인이 된다.

고정된 시간 지연을 갖는 원격 제어 시스템에 대해서는 이미 원격 로보틱스 분야에서 많은 연구가 진행되어 왔다. Anderson[11]은 시간 지연을 갖는 양방향 제어기에 대한 연구를 수행하였고, Yokokohji[10]는 원격 제어 시스템에 적합한 제어 구조를 제안하였다. Mitsuishi[9]는 시간 지연을 갖는 시스템에서 힘뿐만 아니라 소리 정보와 영상 정보도 함께 되먹임 시킬 수 있는 시스템 구조를 제안하였다. 이형기[12]는 모델링 불확실성과 시간 지연이 있는 원격조작기의 강인한 양방향 제어 구조를 제안하였다. 하지만 이와 같이 고정된 시간 지연을 고려한 원격 로보틱스 분야의 많은 연구 성과는 인터넷 로보틱스 분야에 그대로 적용하기 어렵다. 최근에 인터넷 시간 지연을 확률 분포로 모델링하여 과거의 원격 로보틱스 분야의 연구 성과를 인터넷과 접목시키려는 연구가 수행되고 있다. Oboe[2]는 힘을 되먹임 받는 원격 제어 시스템을 인터넷과 접목시킬 수 있는 구조를

제안하였다. 시간 지연을 극복하기 위한 방법으로 로봇에게 보다 많은 지능을 부여하고, 상위 레벨 행동을 제어 입력으로 사용하는 행동 기반 제어기에 대한 연구도 활발히 진행되어 왔다. Stein[14]은 시간 지연을 갖는 원격 제어 시스템에서 행동 기반 제어기를 구현하였고, Cho[15]는 원격 제어 시스템에서 이산 사건 제어 입력을 사용하였다. Tarn [16,3]은 로봇 팔의 제어에 이산 사건의 개념을 적용하였고, 인터넷 시간 지연을 모델링하여 인터넷을 통한 원격 제어 시스템으로 확장시켰다. 하지만 이러한 많은 연구에도 불구하고 인터넷 시간 지연은 여전히 어려운 문제로 남아있다.

Kopacek[19]은 서비스 로봇의 일종인 가정용 로봇의 궁극적인 단계는 퍼스널 로봇이라고 제안한다. Paulos[17]는 인간 중심 로봇의 개념을 제안하고, 인터넷을 통해 다른 사람과 대화를 가능하게 만드는 퍼스널 로봇을 구현하였다. Ota[18]는 스스로 충전할 수 있는 가정용 서비스 로봇을 구현하였다. 하지만 퍼스널 로봇에 대한 연구는 시작 단계라고 볼 수 있다. 또한 퍼스널 로봇과 인터넷의 접목을 위해서는 인간과 로봇간의 인터페이스에 대한 연구도 중요하다. Hirukawa[20]는 인터넷 제어를 위한 인간 지향 인터페이스의 구조를 제안하였고, Backes[5]는 탐사용 로봇을 인터넷으로 제어하기 위한 웹 인터페이스를 구현하였다.

1.3 연구 내용 및 구성

본 논문에서는 인터넷 기반 퍼스널 로봇 시스템의 직접 제어에 적합한 제어 구조를 제안하고자 한다. 원격 제어를 위해 사용자는 가상 환경과 가상 로봇으로 구성된 시뮬레이터를 이용하여 실제 로봇의 위치를 알아내고, 원하는 제어 입력을 생성하게 된다. 사용자에 의해 생성된 로봇의 제어 입력은 인터넷을 통해 로봇으로 인가된다. 제안된 제어 구조는 로봇의 움직임이 인터넷 시간 지연의 영향에 둔감하도록 만든다.

두 가지 측면에서 인터넷 기반 제어 구조에 대한 제안을 한다. 첫 번째는 로봇의 움직임 관점에서의 접근이다. 사용자가 실제 로봇의 정보를 정확히 알기 위해서는 시뮬레이터에 나타난 가상 로봇의 정보와 실제 로봇의 정보가 일치해야 한다. 따라서 두 로봇의 위치와 각도 등을 일치시킬 수 있도록 제어 구조를 제안한다. 두 번째는 가상 환경 관점

에서의 접근이다. 가상 환경은 실제 환경에서 이미 알고 있는 정보만을 나타낼 수 있다. 하지만 실제 환경에 변화가 생길 경우, 가상 환경과 실제 환경은 차이를 보이게 되고, 결과적으로 가상 로봇과 실제 로봇간의 움직임에 차이를 보이게 된다. 이와 같은 문제를 해결하기 위한 제어 구조를 제안한다.

본 논문의 구성은 다음과 같다. 2장에서는 본 논문에서 구상하고 있는 인터넷 기반 퍼스널 로봇 시스템에 대해 설명한다. 퍼스널 로봇의 구조와 전체 시스템의 구조, 그리고 인터넷을 이용하기 위한 인터페이스의 구현 방법에 대해 언급한다. 3장에서는 시뮬레이터에 적용한 가상 로봇과 가상 환경에 대해 설명한다. 가상 로봇의 기구학적 모델과 로봇의 이동 궤적, 센서 모델, 환경 모델 등의 내용을 다룬다. 4장에서는 인터넷 기반 퍼스널 로봇 시스템에 적합한 제어 구조를 제안한다. 먼저 제어 구조를 제안하기 위해 설정한 문제에 대해 언급하고, 장애물 회피 알고리즘을 설명한다. 인터넷 시간 지연의 분포를 조사해 보고, 이론적인 인터넷 시간 지연을 분석한다. 그리고 인터넷 시간 지연이 제어 입력에 미치는 영향에 대해 살펴본다. 인터넷 제어 구조 I, 인터넷 제어 구조 II, 그리고 인터넷 제어 구조 III의 단계를 거쳐 인터넷 기반 제어 구조를 제안하고, 각 구조에 대한 모의 실험을 수행한다. 마지막 절에서는 환경 변화를 가상 환경에 적용할 수 있는 구조를 제안한다. 센서 정보를 이용하여 장애물을 인식하는 방법, 인식된 장애물이 가상 환경에 존재하는지의 여부 판별, 가상 환경에 새로운 장애물을 적용하기 위한 방법 등을 설명하고, 모의 실험을 수행한다. 5장에서는 실제 이동 로봇을 이용하여 제안된 인터넷 제어 구조에 대해 실험하고, 실험 결과를 분석한다. 마지막 6장에서는 결론을 맺고, 앞으로의 과제에 대해 언급한다.

2. 인터넷 기반 퍼스널 로봇 시스템

본 장에서는 인터넷 기반 퍼스널 로봇 시스템의 전체적인 구조에 대해 설명한다. 현재 인터넷 기반 퍼스널 로봇 시스템에 대한 정의가 없기 때문에 본 연구에서 구상하고 있는 시스템에 대한 내용이 중심을 이룬다. 본 장은 세 개의 절로 구성된다. 첫 번째 절에서는 퍼스널 로봇의 구조를 기본 장치와 추가 장치로 구분하고, 두 번째 절에서는 인터넷 기반을 위한 전체 시스템 구조에 대해 다룬다. 그리고 마지막 절에서는 실제로 인터넷을 이용하기 위한 인터페이스의 구현에 대해 크게 두 가지의 방법으로 나누어 설명한다.

2.1 퍼스널 로봇

퍼스널 로봇은 가정에서 인간에게 편의를 제공하는 서비스 로봇의 일종이라고 정의할 수 있다. 퍼스널 로봇은 그 기능에 의해 분류된다. 따라서 하드웨어는 기존의 다른 로봇들과 유사하다. 퍼스널 로봇의 구조는 크게 기본 장치와 추가 장치로 나눌 수 있다. 기본 장치는 임의의 작업을 수행하기 위해서 반드시 있어야할 부분을 말하고, 추가 장치는 특정 작업을 수행하는 경우에 필요한 부분이다. 그림 2.1과 표 2.1은 기본 장치(fundamental device)와 추가 장치(additional device)의 구분을 나타낸다.

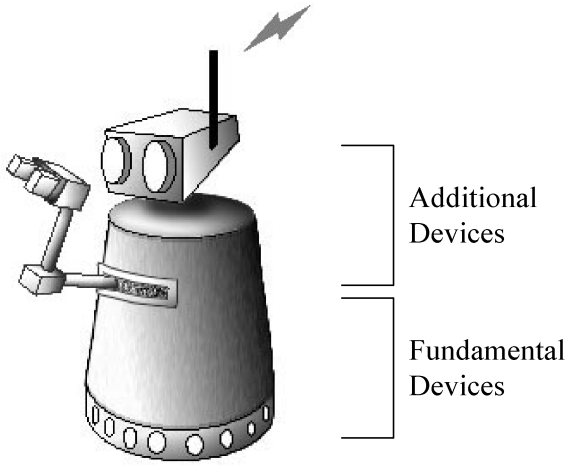


그림 2.1: 퍼스널 로봇의 구조

기본 장치는 로봇이 장애물과의 충돌 없이 원하는 위치로 이동하는 것을 가능하게 하고, 추가 장치는 퍼스널 로봇의 세부 기능을 결정한다. 본 연구에서는 기본 장치만으로 구성된 로봇을 다룬다.

Fundamental devices	Additional devices
■ mobile parts ■ sensors ■ wireless LAN systems ■ personal computer (PC) ■ power systems	■ vision systems ■ robot arms ■ voice systems ■ security systems

표 2.1: 기본 장치와 추가 장치의 분류

2.2 전체 시스템의 구조

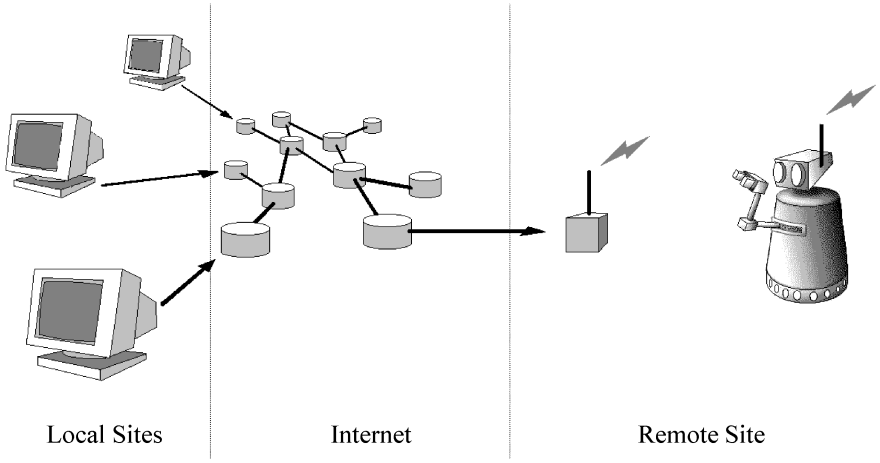


그림 2.2: 인터넷 기반 퍼스널 로봇 시스템

퍼스널 로봇 시스템의 실제 구조는 그림 2.2와 같다. 통신 매개체는 인터넷망을 이용하고, 사용자는 인터넷에 연결되어 있는 임의의 컴퓨터를 이용한다. 로봇 시스템은 무선 랜(LAN) 시스템을 이용하거나, 또는 인터넷과 연결되어 있는 컴퓨터와 무선 통신을 함으로써 인터넷망과의 접속을 이룬다. 독립적으로 동작하는 퍼스널 로봇 시스템에서는 후자의 방법으로 인터넷과 접속하는 것이 바람직하다.

인터넷 기반 퍼스널 로봇 시스템을 구현하는데 있어서 전체 시스템의 원격 제어 구조를 결정하는 일은 상당히 중요하다. 원격 제어 구조는 크게 단방향 제어와 양방향 제어로 나눌 수 있고, 또한 장애물 회피 관점에서 볼 때 이동 로봇과 장애물과의 충돌 방지를 위한 제어기의 구현 여부에 따라 나눌 수 있다. 그림 2.3은 원격 제어 구조를 모두 여섯 가지의 형태로 분류한 것이다.

단방향 제어 구조는 속도 명령을 로봇에게 보낼 수 있지만, 로봇의 현재 상태를 정확히 파악하는 것이 불가능하기 때문에 인터넷 기반 시스템에서 사용하기에는 적합하지 않다. 시뮬레이터에 구현된 로봇의 모델과 실제 로봇의 모델이 정확히 일치할 경우 시뮬레이터의 로봇과 실제 로봇의 상태가 일치하지만, 정확히 일치하는 로봇 모델을 찾아내는 것은 불가능하다. 시뮬레이터 모델과 실제 모델의 차이에 의해 발생하는 오차는 양방향 제어 구조를 적용함으로써 줄일 수 있다.

그림 2.3의 구조(i)의 경우, 사용자가 시뮬레이터 상에서 로봇 주변의 장애물 정보를 파악할 수 있다고 가정할 때, 인터넷에 의한 시간 지연으로 인해 충돌을 완벽하게 방지하는 것은 불가능하다. 또한 사용자의 조종 실수로 인한 충돌도 막을 수 없다. 제어 구조(ii)는 장애물 회피 기능을 로봇 내부의 제어기로 추가 구현한다. 로봇은 초음파 센서를 이용하여 주변 환경의 장애물과의 거리 정보를 알아내고, 그 정보를 이용하여 장애물과 충돌하지 않는 방향으로 진로 방향을 수정해 나간다. 이와 같은 구조는 항상 장애물과의 충돌을 회피하는 것을 가능하게 한다. 제어 구조(iii)는 이미 알려진 로봇 주변 환경의 장애물 정보를 이용하여 시뮬레이터 내부에 장애물 회피 제어기를 추가 구현한다. 마지막 구조는 이미 알려진 장애물이 있는 경우 시뮬레이터의 로봇과 실제 로봇의 행동을 일치시킨다는 장점을 갖는다.

본 논문에서는 그림 2.3의 양방향 제어 구조(b)-(iii)을 이용하여 인터넷 기반 퍼스널 로봇 시스템을 구현한다.

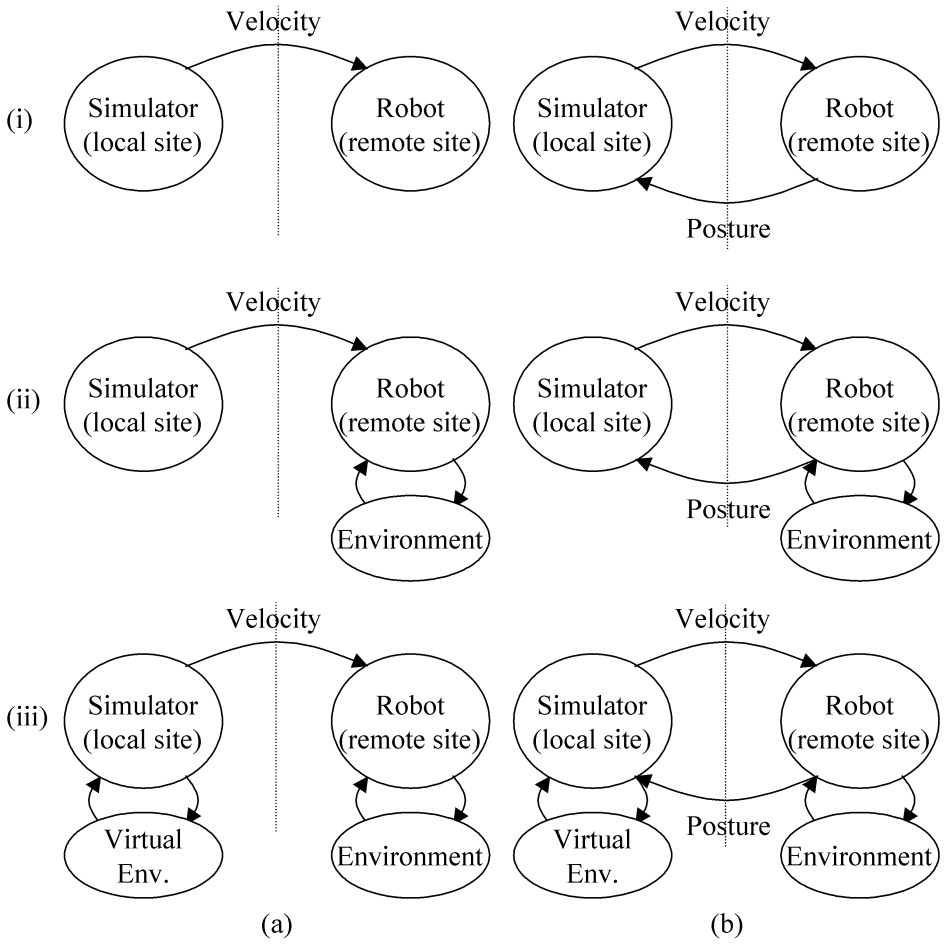


그림 2.3: 원격 제어 구조의 분류

2.3 인터페이스 구현 방법

인터넷을 통하여 로봇과 정보를 주고받기 위해서는 사용자가 사용하기에 적합한 인터페이스가 필요하다. 이때 인터페이스를 실제로 구현하는 방법은 표 2.2와 같이 크게 두 가지 방법으로 나눌 수 있다.

첫 번째는 현재 인터넷에서 가장 많이 사용되고 있는 웹(Web)을 이용하는 방법이다. 로봇이 위치한 곳에 웹 서버를 실행시켜 놓으면 인터넷에 연결되어 있는 임의의 장소에서 웹 브라우저를 이용하여 로봇과 접속을 할 수 있다. 이때 웹 서버와 로봇 제어기는 CGI를 통해 정보를 주고받고, local site에서 필요한 시뮬레이터는 Java를 이용하여 remote site로부터 공급받을 수 있다.

두 번째는 독립적으로 실행되는 서버 프로그램과 클라이언트 프로그램을 C 또는 C++ 등의 언어를 이용하여 직접 구현하는 방법이다. 이 방법은 구현된 서버 프로그램과 클라이언트 프로그램이 직접 TCP/IP 프로토콜을 사용하여 정보를 주고받는다.

두 가지의 방법은 각각 장단점을 갖는다. 첫 번째 방법은 웹 브라우저만을 이용하여 쉽게 로봇 시스템에 접근이 가능하다는 장점이 있다. 하지만 개발자의 입장에서는 까다로운 보안 문제와, CGI, HTML, Java 등의 다양한 프로그래밍 방법의 혼용 등에 따른 번거로움이 있다. 두 번째 방법은 보안 문제와는 무관하며, 하나의 언어를 이용하여 인터페이스를 구현할 수 있다는 장점이 있다. 하지만 로봇 시스템에 접근을 하기 위해서는 먼저 클라이언트 프로그램을 서버로부터 직접 전송 받아야 한다는 단점이 있다.

본 논문에서는 두 번째 방법을 이용하여 인터페이스를 구현한다. 하지만 퍼스널 로봇 시스템에서는 사용자의 편의성을 우선적으로 고려해야 하므로, 첫 번째 방법은 추후 과제로 남겨둔다.

<table>
<tr><td colspan="2"></td><td>Method 1</td><td>Method 2</td></tr>
<tr><td rowspan="3">Remote Site</td><td colspan="3">Robot</td></tr>
<tr><td>Interface</td><td>CGI</td><td rowspan="2">Server Program
(C, C++, etc.)</td></tr>
<tr><td>Server Application</td><td>Web Server</td></tr>
<tr><td>Protocol</td><td colspan="3">TCP/IP</td></tr>
<tr><td rowspan="3">Local Site</td><td>Client Application</td><td>Web Browser</td><td rowspan="2">Client Program
(C, C++, etc.)</td></tr>
<tr><td>Interface</td><td>Java</td></tr>
<tr><td colspan="3">User</td></tr>
</table>

표 2.2: 인터페이스 구현 방법

3. 이동 로봇 및 환경 모델

본 장에서는 기본 장치로 이루어진 이동 로봇의 모델과 시뮬레이터 상의 가상 환경 모델에 대해 설명한다. 첫 번째 절에서는 이동 로봇의 기구학과 제한 조건, 그리고 이동 궤적 계산 방식에 대해 언급한다. 두 번째 절에서는 이동 로봇의 센서 모델에 대해 설명하고, 마지막 절에서는 이동 로봇의 주변 환경을 모델링한 방법에 대해 설명한다. 본 장에서 설명한 모델들은 모두 시뮬레이터를 구현할 때 적용된다.

3.1 이동 로봇 모델

본 절에서는 local site의 시뮬레이터에서 구현된 이동 로봇의 모델과 제한 조건에 대해 설명한다. 이동 로봇의 구조와 제한 조건, 기구학적 모델, 그리고 로봇의 이동 궤적 계산 방식의 순서로 설명한다.

3.1.1 이동 로봇의 기구학적 모델

이동 로봇의 구조는 그림 3.1(a)와 같다. L은 두 바퀴 사이의 거리이며, R은 바퀴의 반지름, W는 로봇의 가로와 세로 길이를 나타낸다. 로봇은 두 개의 바퀴를 가지고 있으며, 바퀴의 회전축은 바닥의 수직선분과 직교한다. 바퀴와 바닥 사이의 접촉면에서 pure rolling과 non-slipping의 조건을 만족한다. Pure rolling은 미끄러짐이 없는 굴림 운동이라는 뜻으로 바퀴 표면의 점들이 연속적으로 바닥 표면의 점들과 일대일 접촉을 할 때를 말한다. Non-slipping은 나아가는 방향의 수직 성분으로 미끄러지지 않는다는 것을 말하며, 로봇은 이동 중 옆으로 미끄러지지 않는다는 것을 의미한다. 로봇은 2차원 평면 위에서 움직인다.

로봇의 기구학적인 모델은 그림 3.1(b)에 의해 설명될 수 있다. 로봇의 중심 위치와 방향각을 모두 합쳐 식(3.1)과 같이 자세(posture) 벡터를 정의한다.

$$\boldsymbol{P}_c = [\, x_c \;\; y_c \;\; \theta_c \,]^T \tag{3.1}$$

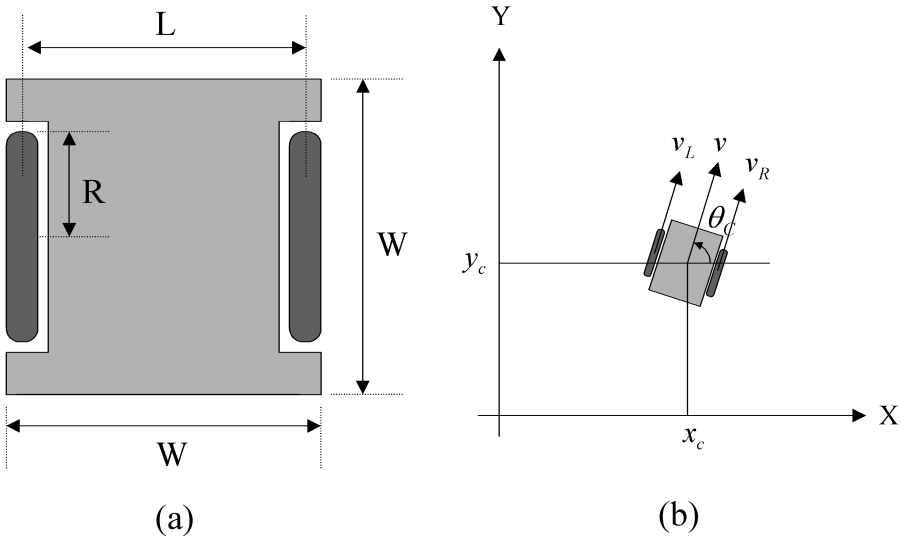


그림 3.1: 이동 로봇의 구조와 2차원 평면에서의 자세

로봇의 왼쪽 바퀴와 오른쪽 바퀴의 각속도를 각각 ω_L, ω_R, 왼쪽 바퀴와 오른쪽 바퀴의 지면과의 접촉점의 선속도를 각각 v_L, v_R, 그리고 로봇의 제어 입력으로 인가되는 로봇 중심의 선속도와 각속도를 각각 v, ω 라고 하면, 각 변수들은 다음과 같은 관계를 갖는다.

$$\begin{bmatrix} v_R \\ v_L \end{bmatrix} = R \begin{bmatrix} \omega_R \\ \omega_L \end{bmatrix}$$
$$\begin{bmatrix} v \\ \omega \end{bmatrix} = \begin{bmatrix} \frac{1}{2} & \frac{1}{2} \\ \frac{1}{L} & -\frac{1}{L} \end{bmatrix} \begin{bmatrix} v_R \\ v_L \end{bmatrix} \tag{3.2}$$

또한 로봇의 자세 벡터 $\boldsymbol{P}_c$ 와 제어 입력 $\boldsymbol{u} = [v\ \omega]^T$는 자코비안 행렬 J 에 의해 다음 식(3.3)과 같은 관계를 가진다.

$$\begin{bmatrix} \dot{x_c} \\ \dot{y_c} \\ \dot{\theta_c} \end{bmatrix} = J(\theta) \begin{bmatrix} v \\ \omega \end{bmatrix} = \begin{bmatrix} \cos\theta & 0 \\ \sin\theta & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v \\ \omega \end{bmatrix} \tag{3.3}$$

3.1.2 이동 로봇의 궤적

로봇의 제어 입력 $[v\ \omega]^T$에 따른 이동 궤적은 식(3.3)을 이용하여 다음과 같이 구할 수 있다. 이때 샘플링 주기는 T 이고, 시간은 $t = kT$,$(k=0,1,2,\cdots)$ 이다.

$$\begin{aligned}\begin{bmatrix} x_c(k) \\ y_c(k) \\ \theta_c(k) \end{bmatrix} &= \begin{bmatrix} x_c(k-1) \\ y_c(k-1) \\ \theta_c(k-1) \end{bmatrix} + T \begin{bmatrix} \Delta x_c \\ \Delta y_c \\ \Delta \theta_c \end{bmatrix} \\ &= \begin{bmatrix} x_c(k-1) \\ y_c(k-1) \\ \theta_c(k-1) \end{bmatrix} + T \begin{bmatrix} \cos\theta_c(k-1) & 0 \\ \sin\theta_c(k-1) & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v \\ \omega \end{bmatrix}\end{aligned} \tag{3.4}$$

이동 로봇의 궤적식(3.4)은 연속 시간 영역(continuous-time domain)에서 적분 형태로 표현 될 때 실제 이동 궤적과의 오차를 없앨 수 있다. 하지만 샘플링 주기 T 값이 커질수록 큰 오차가 나타나게 되어 궤적식으로 사용할 수 없게 된다. 인터넷을 포함한 제어 구조에서의 샘플링 주기 T 값은 상당히 크기 때문에, 본 논문에서는 오차를 줄일 수 있는 궤적식(3.5)를 이용한다.

$$\begin{aligned}\begin{bmatrix} x_c(k) \\ y_c(k) \\ \theta_c(k) \end{bmatrix} &= \begin{bmatrix} x_c(k-1) \\ y_c(k-1) \\ \theta_c(k-1) \end{bmatrix} + \\ &\begin{bmatrix} \cos\theta_c(k-1) & -\sin\theta_c(k-1) & 0 \\ \sin\theta_c(k-1) & \cos\theta_c(k-1) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \frac{v}{\omega}\sin\omega T \\ \frac{v}{\omega}(1-\cos\omega T) \\ \omega T \end{bmatrix}\end{aligned} \tag{3.5}$$

이 때 제어 입력에서 각속도 값이 $\omega = 0$ 의 조건을 갖는 경우는 다음 궤적식(3.6)을 이용하여 계산한다.

$$\begin{bmatrix} x_c(k) \\ y_c(k) \\ \theta_c(k) \end{bmatrix} = \begin{bmatrix} x_c(k-1) \\ y_c(k-1) \\ \theta_c(k-1) \end{bmatrix} + \begin{bmatrix} \cos\theta_c(k-1) & -\sin\theta_c(k-1) & 0 \\ \sin\theta_c(k-1) & \cos\theta_c(k-1) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} vT \\ 0 \\ 0 \end{bmatrix} \tag{3.6}$$

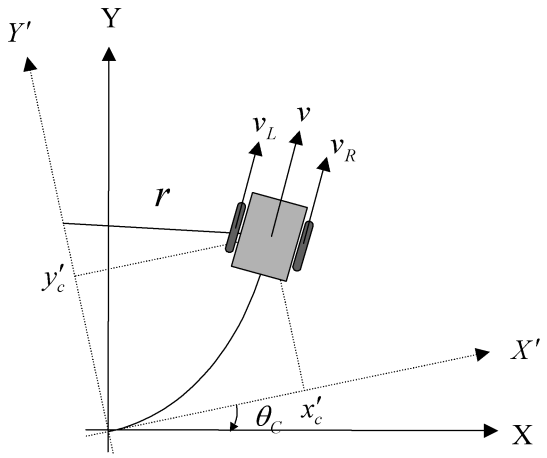


그림 3.2: 로봇의 이동 궤적

로봇의 이동 궤적식(3.5), (3.6)은 그림 3.2에서 간단히 유도될 수 있다. 먼저 로봇의 초기 자세를 $\boldsymbol{P}_c'(0)=[0\ 0\ 0\ ]^T$ 로 갖는 2차원 평면 $X'Y'$ 좌표계를 생각한다. 로봇의 회전 반경을 r 이라고 하면, $r = \frac{v}{\omega}$ 이고, 로봇의 자세(posture)는 다음과 같은 과정으로 구할 수 있다.

$$\boldsymbol{P}_c' = [\ r\sin\omega T \quad r(1-\cos\omega T) \quad \omega T\ ]^T$$

$$\therefore\ \boldsymbol{P}_c(k) = \boldsymbol{P}_c(k-1)+\begin{bmatrix} \cos\theta_c(k-1) & -\sin\theta_c(k-1) & 0 \\ \sin\theta_c(k-1) & \cos\theta_c(k-1) & 0 \\ 0 & 0 & 1 \end{bmatrix}\boldsymbol{P}_c'(k)$$

이 때 $\omega \to 0$ 일 경우, 다음과 같이 정리된다.

$$\lim_{\omega\to 0}\boldsymbol{P}_c' = \lim_{\omega\to 0}\left[\ \frac{v}{\omega}\sin\omega T \quad \frac{v}{\omega}(1-\cos\omega T) \quad \omega T\right]^T = \begin{bmatrix} vT \\ 0 \\ 0 \end{bmatrix}$$

$$\therefore\ \boldsymbol{P}_c(k) = \boldsymbol{P}_c(k-1)+\begin{bmatrix} \cos\theta_c(k-1) & -\sin\theta_c(k-1) & 0 \\ \sin\theta_c(k-1) & \cos\theta_c(k-1) & 0 \\ 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} vT \\ 0 \\ 0 \end{bmatrix}$$

3.2 센서 모델

본 절에서는 장애물 인식을 위한 센서 모델에 대해 설명한다. 센서로 사용할 수 있는 장치로는 적외선 센서나 초음파 센서, 비전 시스템 등을 들 수 있다. 적외선 센서는 센서 간의 간섭이 적다는 장점과 장애물 인식 거리가 짧다는 단점을 갖는다. 초음파 센서는 비교적 먼 장애물에 대해서 적용이 가능하지만 잡음의 영향을 많이 받는다는 단점이 있다. 비전 시스템은 시야 범위 내의 장애물과의 거리, 장애물의 형태 등 다양한 정보를 알아낼 수 있다. 하지만 장애물 인식을 위한 긴 처리 시간과 많은 정보 처리량을 단점으로 들 수 있다. 본 연구에서는 초음파 센서 모델을 사용한다. 초음파 센서 모델은 센서와 장애물과의 정확한 거리를 알아낼 수 있고, 센서간의 간섭을 받지 않는다고 가정한다. 실제 초음파 센서의 경우 최소 해상도에 있어서 한계가 있지만, 본 모델에서는 무시한다.

그림 3.3은 장애물 감지 센서 모델을 나타내고, 표 3.1은 파라메터에 대한 설명이다.

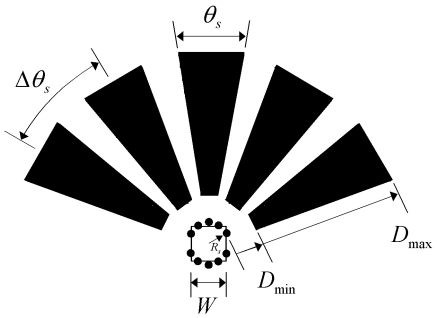


그림 3.3: 장애물 감지 센서

파라메터	의 미	파라메터	의 미
N_s	총 센서의 개수	$D_{\max}$	최대 센싱 거리
θ_s	센싱 시야각	$D_{\min}$	최소 센싱 거리
$\Delta\theta_s$	센서간의 사이각	R_s	로봇 중심과 센서의 거리

표 3.1: 장애물 감지 센서 파라메터

3.3 환경 모델

그림 2.3의 제어 구조(iii)을 이용하여 인터넷 기반 퍼스널 로봇 시스템을 구현하기 위해서는 시뮬레이터에서 사용할 가상 환경의 모델이 필요하다. 퍼스널 로봇 시스템의 경우 동일한 환경 하에서 주로 동작하게 된다. 따라서 이미 알려진 주변 환경에 대한 정보는 local site에서 가상 환경으로 적용될 수 있다. 본 논문에서는 주변 환경에서 장애물에 대한 정보만을 이용하므로, 환경은 다음과 같이 간단히 모델링될 수 있다.

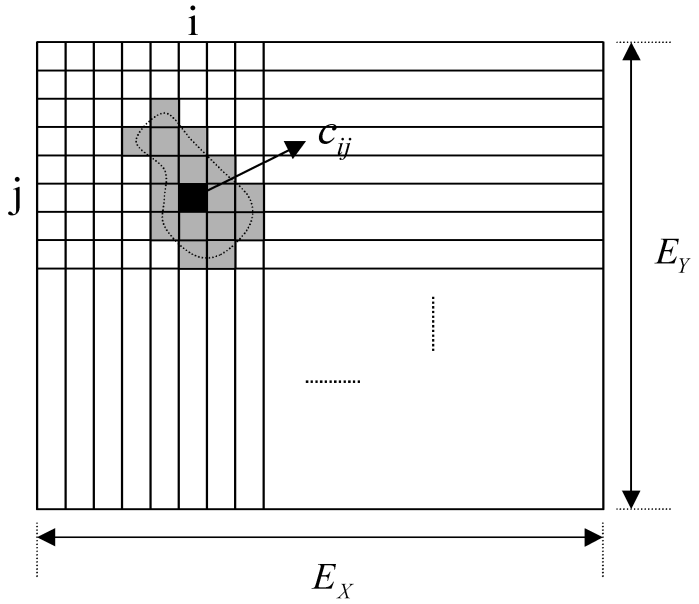


그림 3.4: 장애물 환경 모델링

그림 3.4에서와 같이 로봇이 동작하는 환경의 전체 영역은 $E_X \times E_Y$ 의 크기로 정의되며, 일정 간격의 셀(cell) 단위로 분할된다. 각 셀(cell)의 값은 장애물의 존재 여부에 따라 다음과 같이 정의된다.

$$\begin{cases} c_{ij} = 0 & \text{장애물이 포함되지 않은 경우} \\ c_{ij} = 1 & \text{장애물이 포함된 경우} \end{cases}$$

4. 인터넷 기반 퍼스널 로봇 시스템 구조 설계

본 장에서는 인터넷 기반 퍼스널 로봇 시스템의 전체 구조를 설계한다. 인터넷 기반 퍼스널 로봇 시스템 구현시 가장 큰 문제점은 인터넷에 의해 발생하는 시간 지연이다. 인터넷 시간 지연은 항상 변하며, 거리나 시간, 전송 데이터 량 등의 함수로 정확한 표현이 불가능하다. 예측이 불가능한 인터넷 시간 지연은 인터넷 기반 퍼스널 로봇 시스템의 동작에 가장 큰 영향을 미친다. 첫 번째 절에서는 인터넷 기반 퍼스널 로봇 시스템에 적합한 동작 조건을 고려하여 전체 시스템 설계시 고려해야할 문제들을 설정한다. 두 번째 절에서는 장애물 회피 알고리즘에 대해 다룬다. 세 번째 절에서는 인터넷 시간 지연의 특성에 대해서 알아보고, 네 번째 절에서는 인터넷 시간 지연 특성의 영향을 받지 않고 첫 번째 절에서 설정한 문제들을 해결할 수 있는 시스템 구조를 단계적으로 제안한다. 이때 제안한 각 인터넷 제어 구조에 대한 모의 실험을 수행하고, 모의 실험 결과를 분석한다.

4.1 문제 설정

로봇의 움직임 제어는 실제 로봇과 거리가 떨어져 있는 위치의 사용자에 의해 이루어진다. 사용자는 로봇의 주변 상황을 실시간으로 파악하는 것이 불가능하므로, 로봇이 장애물과 충돌할 가능성은 매우 높다. 장애물과 충돌할 경우 로봇은 손상을 입을 수 있고, 로봇의 주변에 있는 사람에게 상해를 입힐 수 있다. 인터넷 기반 퍼스널 로봇 시스템의 구현에서 장애물 회피는 매우 중요하다. 로봇은 장애물이 있을 경우 원격 제어 입력을 따르지 않고 스스로 충돌을 회피할 수 있는 능력을 갖추어야 한다.

사용자는 시뮬레이터에 나타난 로봇의 움직임을 실제 로봇의 움직임으로 생각한다. 시뮬레이터의 로봇과 실제 로봇의 동작에 있어서 큰 차이를 보인다면 사용자는 퍼스널 로봇 시스템을 신뢰할 수 없다. 사용자는 실제 로봇의 움직임을 전혀 고려하지 않고 시뮬레이터만을 통해 제어를 할 수 있는 시스템이 가장 이상적이다. 따라서 시뮬레이터의 가상 로봇의 궤적과 실제 로봇의 궤적의 오차를 줄여야 한다.

가상 로봇과 실제 로봇의 궤적이 일치할 경우, 두 로봇의 움직임에 큰 시간차가 존재한다면 사용자는 큰 불편을 느끼게 된다. 실제 로봇이 사용자에 의한 수 분전의 원격 제어 입력을 현재 수행하고 있다면 사용자는 실제 로봇이 가상 로봇의 움직임을 따라올 때까지 기다려야만 할 것이다. 가상 로봇과 실제 로봇의 움직임 사이에 존재하는 시간차를 최소화하는 작업이 필요하다.

인터넷 기반 퍼스널 로봇 시스템을 구현하는데 있어서 고려해야할 문제들은 다음과 같은 항목으로 정리될 수 있다.

▌ 문제 설정

① 로봇의 장애물 회피 기능 보장

② Local site와 remote site의 두 로봇 궤적의 오차 최소화

③ Local site와 remote site의 두 로봇의 움직임 시간차 최소화

그림 4.1은 문제 설정의 ②, ③에서 언급한 local site와 remote site 두 로봇간의 궤적 오차와 움직임의 시간차를 나타낸다. 그림 4.1(b)의 *command* 는 로봇에 인가되는 제어 입력을 말한다.

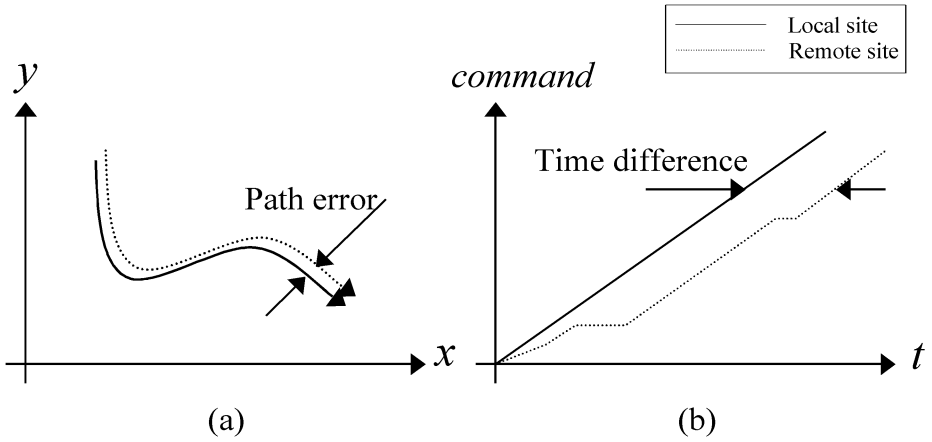


그림 4.1: 로봇의 궤적 오차와 움직임 시간차

4.2 장애물 회피

항상 장애물과의 충돌 회피를 보장하기 위해서는 원격 제어 구조를 그림 3.3의 (iii)과 같이 구현해야 한다. 원격 제어 입력 값과는 무관한 로봇 자체의 장애물 회피 제어기를 포함해야 한다. 그림 4.2는 장애물 회피 기능을 갖춘 로봇 내부의 블럭도를 나타낸다.

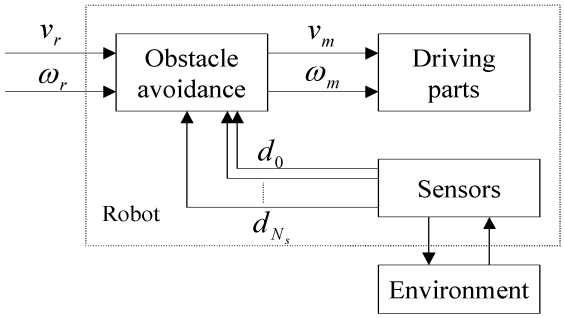


그림 4.2: 로봇의 내부 블럭 구조

장애물 회피 제어기에 대해서는 이미 많은 연구[26,27]가 이루어졌다. 본 논문에서는 중력장 개념을 적용하여 간단히 구현한다. 장애물과 로봇의 최소 거리 $D_{\min}$, 최소 거리에 도달하기 전에 로봇이 정지할 수 있는 안전 거리 D_s 를 정의한다. 로봇의 최대 속도는 $v_{\max}$ 이고, 로봇의 최대 가속도는 $a_{\max}$, 로봇의 질량은 M_R 이다. 로봇의 제어 입력 v_r, ω_r 이 주어질 경우, 로봇의 센서 입력 값 $d_0, d_1, \cdots, d_{N_s}$ 을 이용한 최종 입력 값 v_m, ω_m 은 다음과 같이 구할 수 있다.

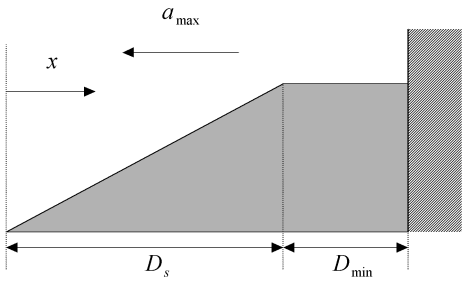


그림 4.3: 장애물의 에너지 레벨

$$\frac{1}{2} M_R v^{*2} + M_R a_{\max} x = M_R a_{\max} D_s$$
$$\therefore\ v^* = \sqrt{2a_{\max}(D_s - x)} \tag{4.1}$$

where

$$D_s = \frac{1}{2} a_{\max} t_{\max}^2 = \frac{v_{\max}^2}{2a_{\max}}$$

$$\therefore\ v_m = \begin{cases} v_r & , \text{ if } x \le 0 \\ sgn(v_r)\ \min(|v_r|, v^*) & , \text{ if } x > 0 \end{cases}$$

where

$$v^* = \begin{cases} \sqrt{2a_{\max}(D_s - x)} & , \text{ if } x \le D_s \\ 0 & , \text{ if } x > D_s \end{cases} \tag{4.2}$$

$$x = (D_s + D_{\min}) - (d_s + R_s)$$

$$d_s = \min(d_0, d_1, \cdots, d_{N_s})$$

식(4.1)은 안전 거리 내에서 로봇과 장애물의 최소 거리를 유지시키기 위한 x 에 따른 속도를 나타내며, 식(4.2)는 로봇의 제어 입력과 장애물 회피를 위한 최종 제어 입력과의 관계를 나타낸다. d_s 는 로봇으로부터 장애물까지의 거리이며, R_s 는 로봇의 중심과 센서와의 거리를 나타낸다.

4.3 인터넷 시간지연

인터넷 시간 지연은 주로 정보가 전송되는 경로 내의 node 수와 각 node에 걸린 부하량에 의해 결정된다. 접속된 두 지역의 물리적인 거리는 직접적인 관련은 없다. 하지만 물리적인 접속 거리가 멀어질수록 두 지역 사이의 node 수도 증가하므로 일반적으로 거리에 따라 인터넷 시간 지연이 증가하는 것처럼 관찰된다. 인터넷 시간 지연은 node의 부하에 의한 영향을 받기 때문에 사용하는 시간대에 따라 큰 차이를 보인다. 밤에는 인터넷 사용자가 적기 때문에 인터넷 시간 지연이 작고, 낮에는 사용자가 증가하여 시간 지연도 커진다. 본 절에서는 이와 같은 인터넷 시간 지연의 분포에 대해 살펴보고, 시스템의 구현에 있어서 가장 큰 문제를 발생시키는 인터넷 시간 지연의 영향을 분석한다.

4.3.1 인터넷 시간 지연의 분포

인터넷 시간 지연을 실제로 측정하여 분포의 특성을 알아보고, 인터넷 시간 지연을 결정하는 요소들에 대해 알아본다. 본 절에서 local site와 remote site로 설정한 지역(same country)은 시뮬레이션과 실험에서 사용한 지역과 동일하다.

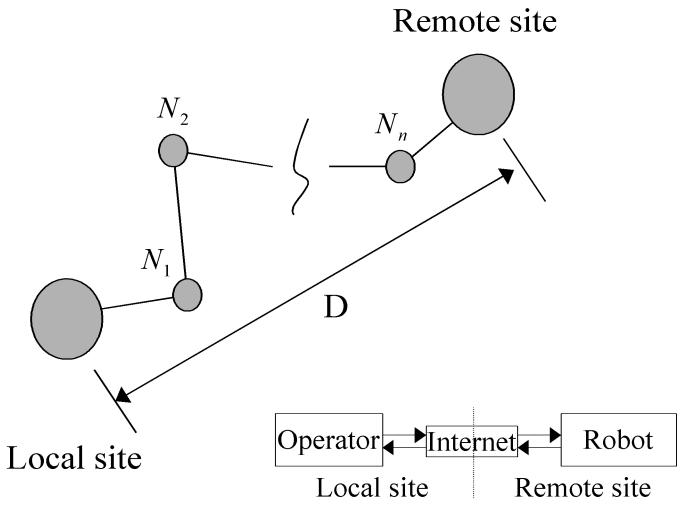


그림 4.4: 인터넷을 이용한 연결 구조

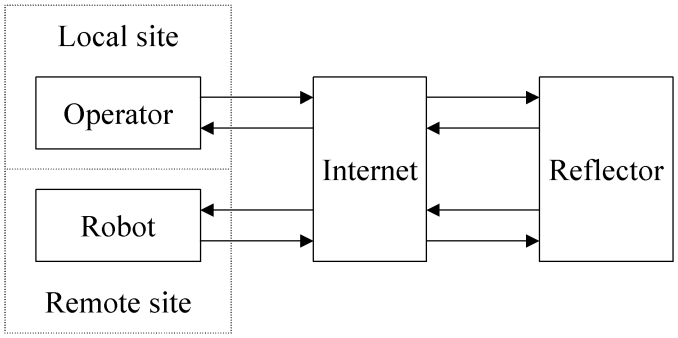


그림 4.5: 반사기가 추가 구성된 인터넷 연결 구조

실제 시스템의 경우 그림 4.4와 같이 local site와 remote site는 물리적인 거리 D 만큼 떨어져 있다. 이와 같은 두 지역을 이용해 실험하는 것에는 많은 어려움이 따른다. Local site와 remote site는 동일한 지역이어야 실험이 가능하다. 따라서 실제 실험은 그림 4.5와 같이 $D/2$ 만큼 떨어진 지역에 반사기(reflector)를 부착하여 한 지역에서 두 site를 모두 구현할 수 있도록 구성한다.

인터넷 시간 지연은 두 지역에 대해서 측정한다. 첫 번째 지역은 같은 도메인(domain)을 갖고, 두 번째는 다른 도메인을 갖는 국내 지역이다. 다음은 실제로 사용한 인터넷 연결 구조의 매개변수 값들을 나타낸다.

	same domain	same country
Distance D	300 m	30 Km
Number of nodes n	11 nodes	29 nodes
Bandwidth b_0, b_n	10 Mbps	10 Mbps
Bandwidth $b_{n/2-1}$, $b_{n/2}$	10 Mbps	56 Kbps

표 4.1: 인터넷 연결 구조의 매개변수 값 설정

그림 4.6과 그림 4.8은 하루 동안의 인터넷 시간 지연 변화를 1분 간격으로 측정한 결과를 나타낸다. 측정한 결과에서 보면 인터넷의 사용 빈도가 높은 오후 시간대에 상당히 큰 시간 지연을 보이고 있음을 알 수 있다. 인터넷의 사용 빈도가 상대적으로 낮은 새벽

4시에서 새벽 7시경 사이는 시간 지연이 가장 작게 나타난다. 시간대에 따른 시간 지연은 유사하지만, 같은 도메인을 갖는 지역의 인터넷 시간 지연이 상당히 작다.

그림 4.7과 그림 4.9는 하루 동안의 인터넷 시간 지연 분포를 나타낸다. 대부분의 경우 시간 지연은 첫 번째 지역의 경우 7ms에서 50ms, 두 번째 지역의 경우 50ms에서 100ms 사이의 값을 갖는다. 하지만 두 번째 지역의 경우 시간 지연이 100ms 보다 큰 값을 갖는 경우에도 일정한 분포를 나타낸다.

그림 4.10에서 그림 4.14은 요일별로 오전 10시부터 오후 10시경까지 인터넷 시간 지연을 측정한 결과이다. 그림 (a)는 시간에 따른 시간 지연을 나타내며, 그림 (b)는 시간 지연의 분포를 나타낸다. 결과에서 요일에 따라 인터넷 시간 지연의 분포가 큰 차이를 보이고 있음을 알 수 있다.

이와 같은 특징은 인터넷 시간 지연을 특정한 확률 분포로 가정하는 것을 어렵게 만든다. 시간 지연을 확률 분포로 가정한 연구[2]가 있지만, 실제와는 많은 차이를 보인다. 표 4.3는 요일별 인터넷 시간 지연의 평균, 표준편차, 최소값 그리고 최대값을 나타낸다.

	mean	standard dev.	min. value	max. value
same domain	20 msec	51.7	6 msec	0.89 sec
same country	4194 msec	2.25×10^4	39 msec	343 sec

표 4.2: 지역별 인터넷 시간 지연의 결과 분석

	mean	standard dev.	min. value	max. value
MON.	531 msec	1.42×10^3	48 msec	15 sec
TUE.	6880 msec	2.89×10^4	45 msec	343 sec
WED.	1480 msec	7.77×10^3	44 msec	154 sec
THU.	1030 msec	4.16×10^3	45 msec	68 sec
FRI.	779 msec	2.59×10^3	49 msec	35 sec

표 4.3: 요일별 인터넷 시간 지연의 결과 분석

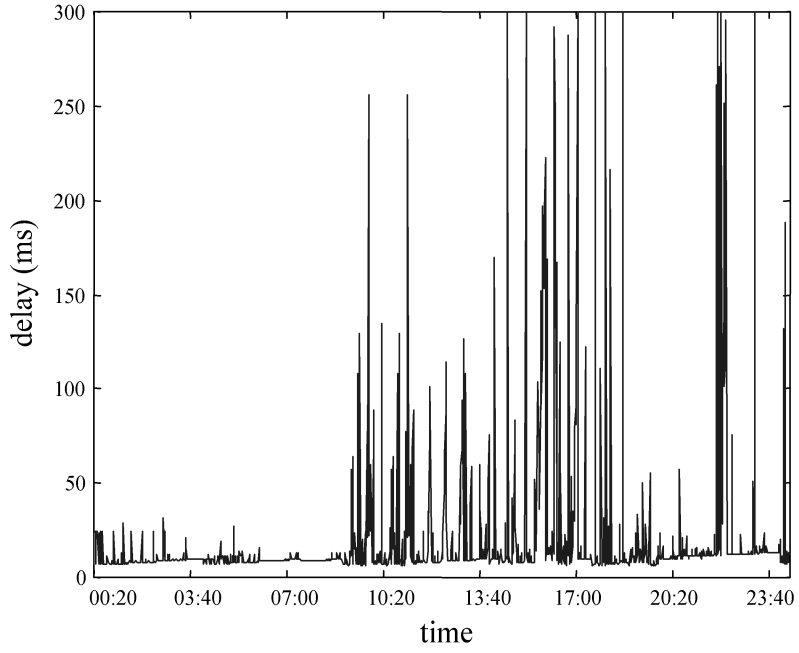


그림 4.6: 1일 동안의 인터넷 시간 지연 (same domain)

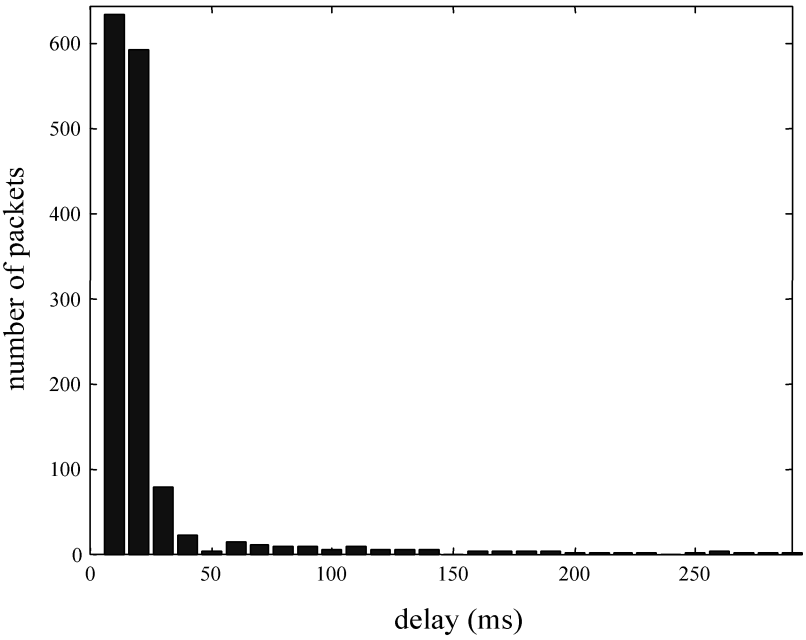


그림 4.7: 1일 동안의 인터넷 시간 지연 분포 (same domain)

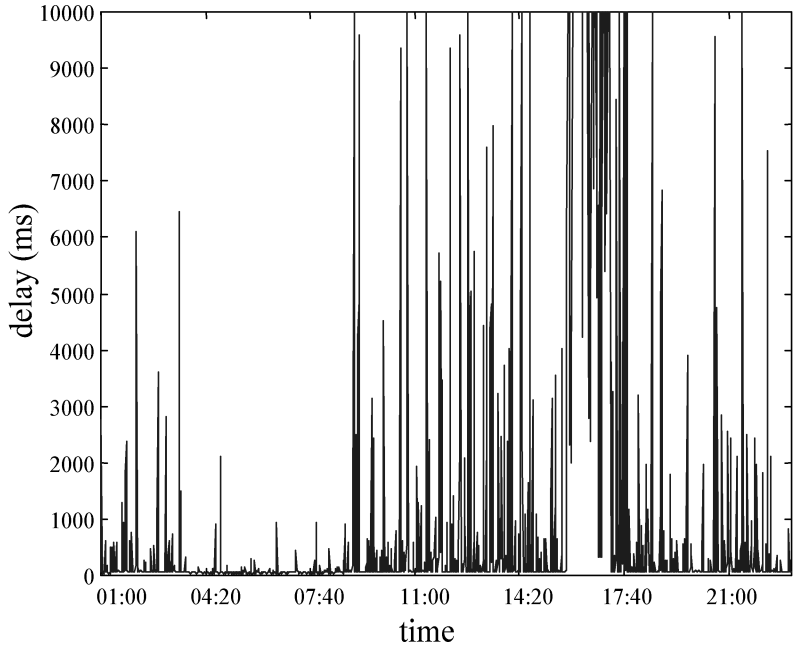


그림 4.8: 1일 동안의 인터넷 시간 지연 (same country)

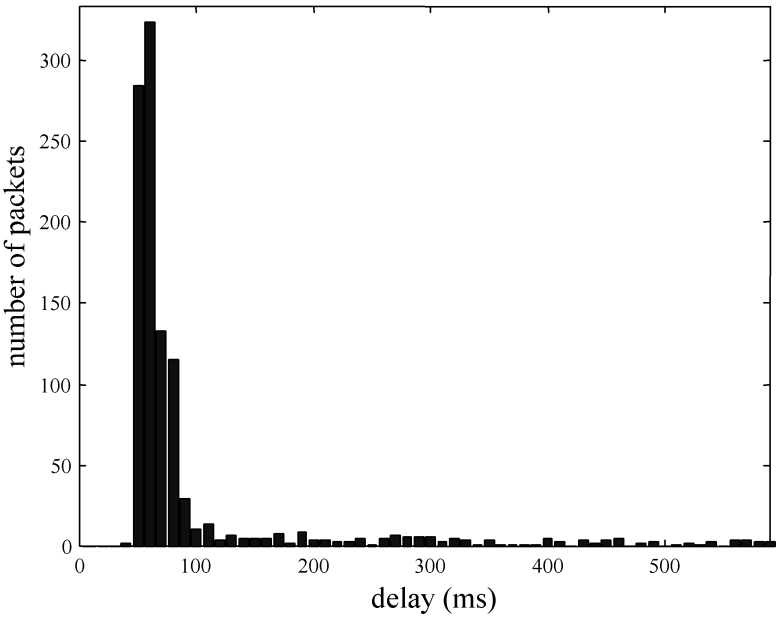
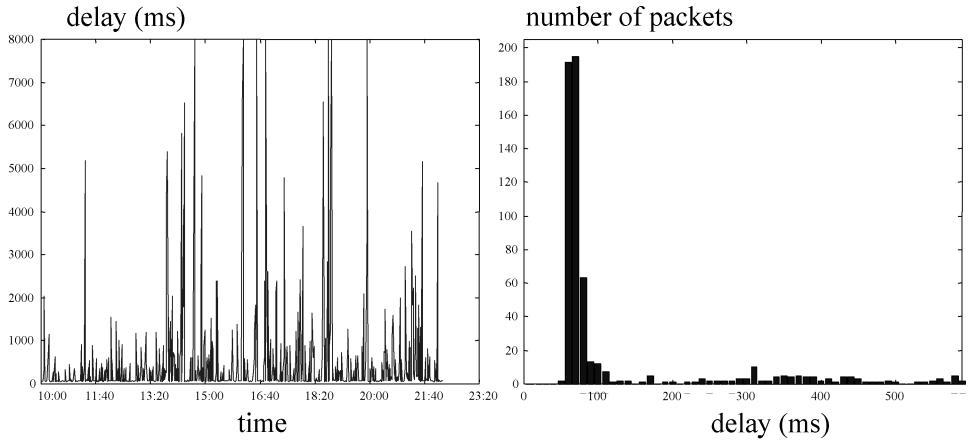
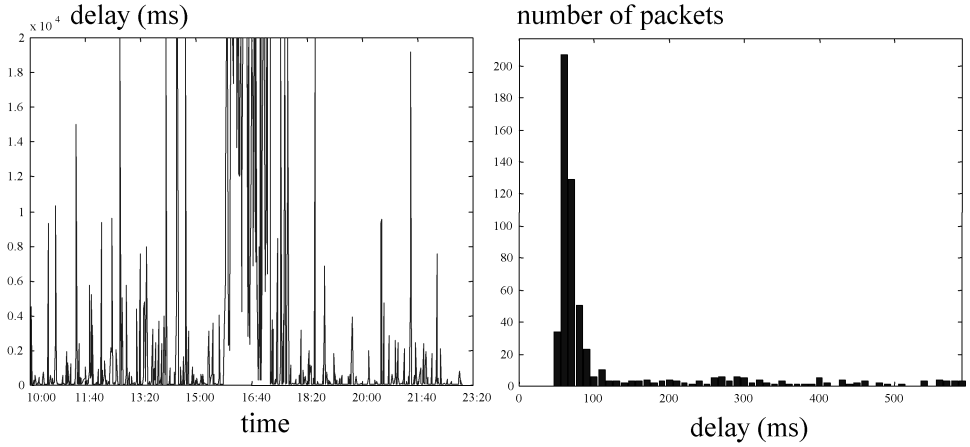


그림 4.9: 1일 동안의 인터넷 시간 지연 분포 (same country)



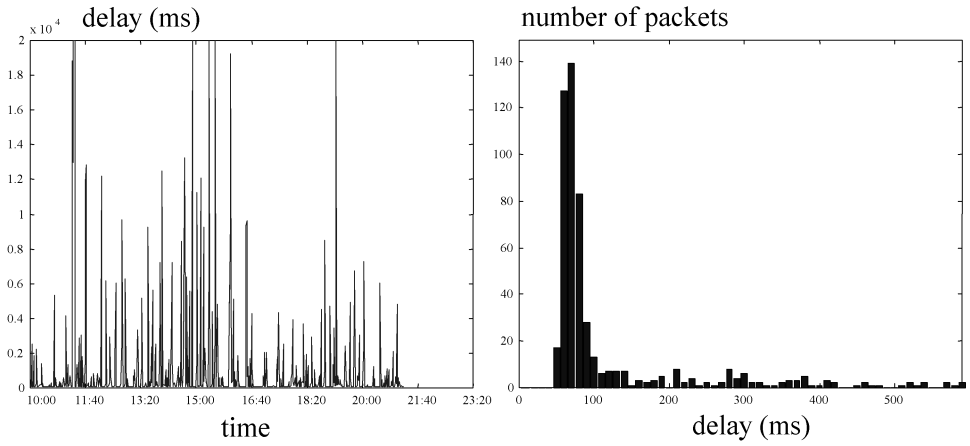
(a) time - delay (b) delay distribution

그림 4.10: 인터넷 시간 지연 분포 (월요일)



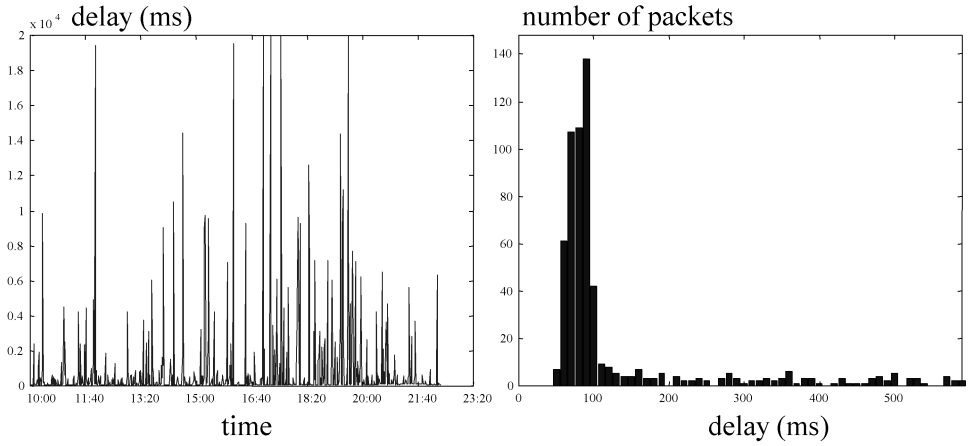
(a) time - delay (b) delay distribution

그림 4.11: 인터넷 시간 지연 분포 (화요일)



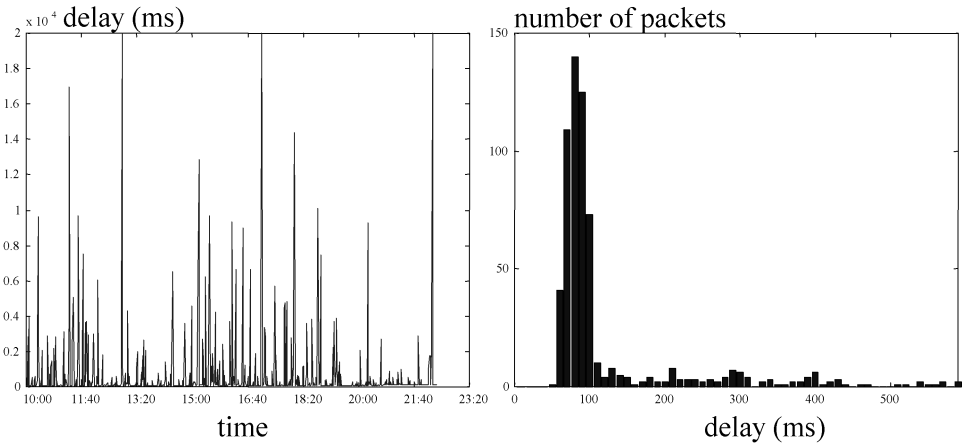
(a) time - delay (b) delay distribution

그림 4.12: 인터넷 시간 지연 분포 (수요일)



(a) time - delay (b) delay distribution

그림 4.13: 인터넷 시간 지연 분포 (목요일)



(a) time - delay (b) delay distribution

그림 4.14: 인터넷 시간 지연 분포 (금요일)

4.3.2 인터넷 시간 지연의 분석

인터넷 시간 지연은 다음과 같은 몇 가지 요소에 의해 결정된다.

① 통신 선로의 정보 전송 속도

② 각 node에서의 정보 처리 속도

③ 각 node의 부하에 따른 지연 시간

④ 통신 선로의 bandwidth와 전송 정보량

대부분의 통신 선로는 광케이블이 사용되므로 정보 전송 속도는 빛의 속도와 같다. 각 node에서의 정보 처리 속도는 node의 컴퓨터 성능에 따라 좌우된다. 각 node의 부하는 시간에 따라 변하고, 확률 분포로 표현이 힘들기 때문에 예측하기가 어렵다. 대체로 인터넷의 이용자 수가 많아지면 시간 지연도 증가한다는 경향을 이미 확인했다. 마지막으로 통신 선로의 bandwidth가 작아지거나, 또는 전송 정보량이 증가하면 시간 지연이 증가하게 된다. 인터넷 시간 지연 $T_d(k)$는 위의 네 가지 요소를 이용하여 다음과 같은 수식으로 표현이 가능하다. 이 때 l_i은 i 번째 통신 선로 길이이고, C는 빛의 속도, t_i^R는 i 번째 node의 라우팅 시간, $t_i^L(k)$는 node의 부하에 의한 시간 지연, M 은 전송 정보량, b_i은 i 번째 통신 선로의 bandwidth를 나타낸다.

$$T_d(k) = \sum_{i=0}^{n} \left[\frac{l_i}{C} + t_i^R + t_i^L(k) + \frac{M}{b_i} \right] \tag{4.3}$$

식(4.3)을 시간에 따라 변하지 않는 항 d_N 과 시간에 따라 변하는 항 $d_L(k)$로 구분하여 다음과 같이 정리할 수 있다.

$$\begin{aligned} T_d(k) &= \sum_{i=0}^{n} \left(\frac{l_i}{C} + t_i^R + \frac{M}{b_i} \right) + \sum_{i=0}^{n} t_i^L(k) \\ &= d_N + d_L(k) \end{aligned} \tag{4.6}$$

4.3.3 인터넷 시간 지연의 영향

인터넷 시간 지연이 시스템의 원격 제어 입력에 미치는 영향을 알아보기 위해 local site에서 sine 값을 인가하여 remote site에 도달하는 파형을 측정한다. 입력 값으로 사용한 sine 함수는 다음과 같다. 주파수 $f=10Hz$, 진폭 $A=5$, 상수 $c=5$, 그리고 샘플링 주기 $T=0.5\sec$ 이다.

$$\begin{aligned} y(k) &= A\ \sin(\ 2\pi f\ kT\ )+c \\ &= 5\ \sin(\ 10\pi\ k\ )+5 \end{aligned} \tag{4.5}$$

그림 4.15는 local site에 입력으로 인가한 값과 remote site에서 받은 값을 동시에 나타낸 것이다. 입력 값은 인터넷 시간 지연의 영향으로 많은 왜곡이 일어남을 알 수 있다. 가장 큰 문제점은 sine 형태의 입력 정보가 손실되는 부분이 존재하게 된다는 것이다. 이것은 인터넷 딜레이가 샘플링 주기보다 크게 될 경우, 이후 두 개 이상의 샘플링 값이 동시에 remote site에 도달하게 되기 때문이다.

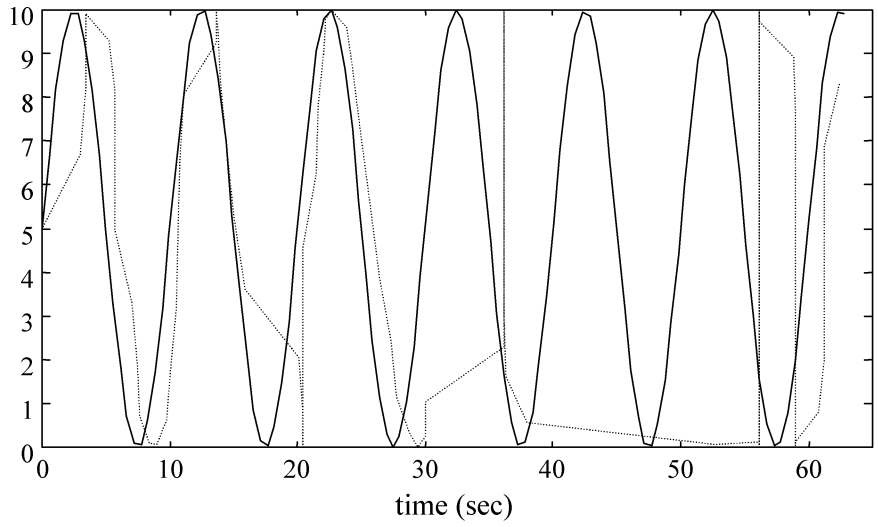


그림 4.15: Sine입력에 대한 시간 지연의 영향

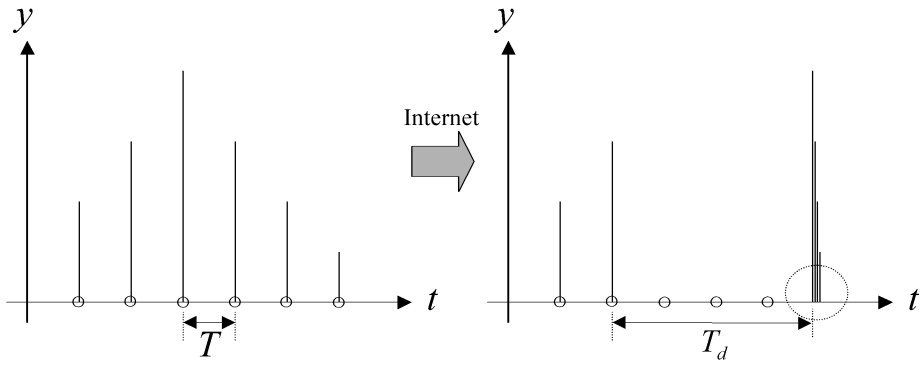


그림 4.16: 인터넷 시간 지연의 영향

그림 4.16은 인터넷 시간 지연이 제어 입력에 미치는 영향을 나타낸다. 샘플링 주기 T 보다 큰 인터넷 시간 지연 T_d 가 존재할 경우, 시간 지연 이후 N 개의 제어 입력이 동시에 remote site에 도달하게 되어 시간에 따른 정보를 잃어버리게 된다.

$$N = \frac{T_d}{T} \tag{4.6}$$

이와 같은 인터넷 시간 지연의 특징은 시간 영역에서 원격 제어를 하는 경우 문제를 발생시키는 가장 큰 원인이 된다. 이것은 한 node에서 많은 부하가 형성되면서 정보가 쌓이기 때문에 나타나는 현상이다. 누적된 정보는 node의 부하가 감소하면서 순간적으로 모두 처리되어 동시에 다음 node로 전송이 이루어진다. 샘플링 주기 T 값이 인터넷 시간 지연 T_d 에 비해 상대적으로 크다면, 그림 4.16과 같은 문제는 발생하지 않는다.

4.4 전체 시스템 구조 설계 및 모의 실험

본 절에서는 인터넷 기반 퍼스널 로봇 시스템의 제어 구조를 설계한다. 앞서 설정한 문제들을 해결해 나가는 방향으로 단계적인 설계가 이루어진다. 전체 시스템은 인터넷 제어 구조 I, 인터넷 제어 구조 II 그리고 인터넷 제어 구조 III의 단계를 거쳐 완성된다.

다음은 각 인터넷 제어 구조에 대한 설명과 모의 실험 결과, 그리고 각 인터넷 제어 구조의 문제점 등을 다룬다. 이때 모의 실험 결과는 시뮬레이터에서 사용하는 가상 로봇 모델과 실제 로봇 모델이 정확히 일치하고, 실제 로봇의 위치는 정확히 안다고 가정한다. 또한 실제 인터넷망의 시간 지연을 이용한다.

4.4.1 인터넷 제어 구조 I

전체 시스템의 기본적인 원격 제어 구조는 이미 2.2절에서 다루었다. 그림 2.3의 원격 제어 구조(b)-(iii)을 기본 구조로 이용하여 설계한다. Remote site의 로봇을 원격 제어하기 위해 local site에는 시뮬레이터를 구현하고, 사용자는 시뮬레이터를 통해 로봇으로 속도 명령을 보낸다. 로봇은 현재의 자세를 시뮬레이터로 되먹임(feedback)시켜 폐루프(closed-loop)를 형성한다. 로봇은 주변 환경을 센싱하여 장애물과의 충돌을 회피하고, 시뮬레이터의 가상 로봇은 이미 알고 있는 가상 환경으로부터 센싱 정보를 받아들여 장애물 회피의 여부를 사전에 파악한다.

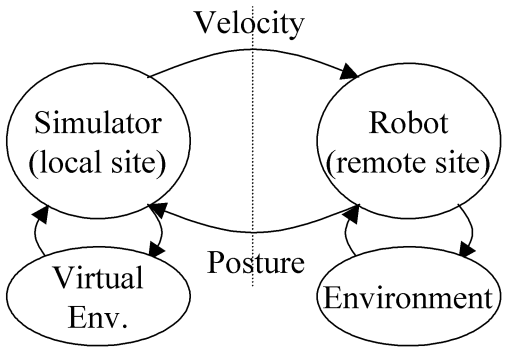


그림 4.17: 전제 시스템의 기본 구조

그림 4.17의 전체 시스템의 기본 구조를 기반으로 설계한 인터넷 제어 구조 I은 그림 4.18와 같다. 사용자 인터페이스(User Interface)는 사용자가 퍼스널 로봇 시뮬레이터(Personal Robot Simulator)의 가상 로봇의 움직임을 시각적으로 관찰하고, 직접 로봇의 제어 입력을 샘플링 주기로 생성하는 기능을 갖는다. 샘플링 주기로 생성되는 로봇의 제어 입력은 인터넷(Internet)망을 통해 로봇으로 입력되고, 동시에 시뮬레이터와 자세 추정기(Posture Estimator)로 입력된다. 로봇은 제어 입력을 받아 움직인 후, 로봇의 바뀐 자세 정보를 인터넷망을 통해 local site의 자세 추정기로 되먹임 시킨다. 자세 추정기(Posture Estimator)는 로봇으로부터 전달된 로봇의 자세 정보를 이용하여 현재 시뮬레이터(Personal Robot Simulator)의 자세를 추정하고, 추정된 값을 시뮬레이터로 입력시킨다. 시뮬레이터는 사용자 인터페이스(User Interface)로부터 제어 입력을 직접 받아 동작하며, 자세 추정기로부터 현재 자세의 추정치를 받아 가상 로봇의 자세를 보정한다. 또한 현재 가상 로봇의 자세는 사용자 인터페이스로 전달되어 시각적인 출력 값으로 사용된다. 실제 로봇과 가상 로봇은 각각 센서를 이용하여 실제 환경과 가상 환경을 통해 주변의 장애물 정보를 알아낸다.

시뮬레이터의 가상 로봇의 자세를 추정하는 자세 추정기(Posture Estimator)는 로봇의 움직임 궤적식(3.5), (3.6)을 이용하여 구현한다. 로봇으로부터 전달되는 로봇의 자세를 $\boldsymbol{P}_c^d(j)$로 정의하면, $(j+1)$ 번째 로봇의 자세 $\widehat{\boldsymbol{P}_c}(j+1)$는 j 번째 로봇의 자세 $\boldsymbol{P}_c^d(j)$와 궤적식을 이용하여 다음과 같은 observer 식으로 구성할 수 있다.

$$\widehat{\boldsymbol{P}_c}(j+1) = \widehat{\boldsymbol{P}_c}(j) + \boldsymbol{J}(\theta_c^d(j))\,\boldsymbol{H}(v_r(j), \omega_r(j)) + \boldsymbol{K}_e(\boldsymbol{P}_c^d(j) - \widehat{\boldsymbol{P}_c}(j))$$

where

$$\boldsymbol{J}(\theta_c) = \begin{bmatrix} \cos\theta_c & -\sin\theta_c & 0 \\ \sin\theta_c & \cos\theta_c & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$\boldsymbol{H}(v_r, \omega_r) = \begin{cases} \begin{bmatrix} \frac{v_r}{\omega_r}\sin\omega_r T & \frac{v_r}{\omega_r}(1-\cos\omega_r T) & \omega_r T \end{bmatrix}^{\mathrm{T}} & , \text{ if } \omega_r \neq 0 \\ [\, T v_r \;\; 0 \;\; 0\,]^{\mathrm{T}} & , \text{ if } \omega_r = 0 \end{cases}$$

로봇의 현재 자세 추정치 $\widehat{P_c}(i)$는 다음과 같이 정리된다. 단, $i > j$ 의 조건을 만족한다.

$$\begin{aligned}
\widehat{P_c}(j+1) &= \widehat{P_c}(j) + J(\theta_c^d(j))\,H(v_r(j),\omega_r(j)) + K_e(P_c^d(j) - \widehat{P_c}(j)) \\
\widehat{P_c}(j+2) &= \widehat{P_c}(j+1) + J(\widehat{\theta_c}(j+1))\,H(v_r(j+1),\omega_r(j+1)) \\
&= (I - K_e)\,\widehat{P_c}(j) + J(\theta_c^d(j))\,H(v_r(j),\omega_r(j)) + K_e P_c^d(j) \\
&\quad + J(\widehat{\theta_c}(j+1))\,H(v_r(j+1),\omega_r(j+1)) \\
&\vdots \quad \vdots \\
\widehat{P_c}(i) &= \widehat{P_c}(i-1) + J(\widehat{\theta}_c(i-1))\,H(v_r(i-1),\omega_r(i-1))
\end{aligned}$$

$$\begin{aligned}
&\therefore \text{ if } \quad i = (j+1)\ ; \\
&\widehat{P_c}(i) = (I - K_e)\,\widehat{P_c}(j) + J(\theta_c^d(j))\,H(v_r(j),\omega_r(j)) + K_e P_c^d(j) \\
&\text{if } \quad i > (j+1)\ ; \\
&\widehat{P_c}(i) = (I - K_e)\,\widehat{P_c}(j) + J(\theta_c^d(j))\,H(v_r(j),\omega_r(j)) + K_e P_c^d(j) \\
&\qquad + \sum_{k=j+1}^{i-1} J(\widehat{\theta_c}(k))\,H(v_r(k),\omega_r(k))
\end{aligned} \tag{4.7}$$

이때 $0 \le K_e \le 1$ 의 조건을 갖는다. 로봇의 자세 추정기는 식(4.7)을 이용하여 가상 로봇의 현재 자세를 알아낼 수 있다. 인터넷 제어 구조 I의 자세 추정기와 인터넷 제어 구조 II, 인터넷 제어 구조 III의 자세 추정기(Posture Estimator)는 모두 동일한 구조를 갖는다.

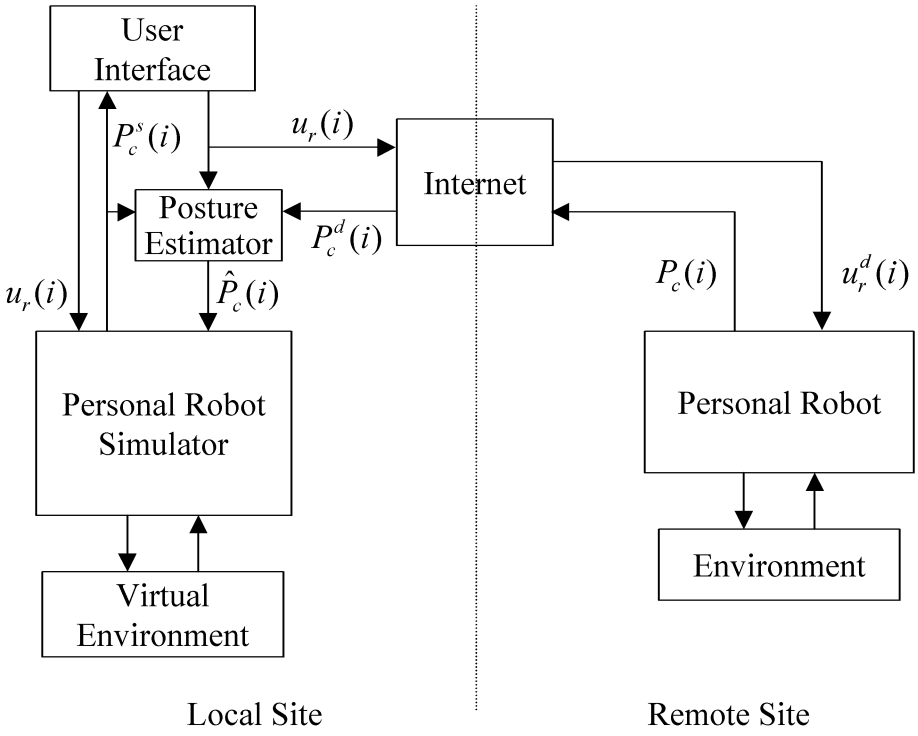


그림 4.18: 인터넷 제어 구조 I

$\boldsymbol{u}_r(i)$: 사용자에 의한 i 번째 제어 입력 $[v_r(i)\ \omega_r(i)]^T$

$\boldsymbol{u}_r^d(i)$: 인터넷망을 통과한 i 번째 제어 입력

$\boldsymbol{P}_c(i)$: 되먹임을 위한 i 번째 로봇 자세

$\boldsymbol{P}_c^d(i)$: 인터넷망을 통과한 i 번째 로봇 자세

$\widehat{\boldsymbol{P}_c}(i)$: 되먹임 값을 이용한 i 번째 로봇 자세 추정치

$\boldsymbol{P}_c^s(i)$: 가상 로봇의 i 번째 로봇 자세

인터넷 제어 구조 I의 모의 실험

인터넷 제어 구조 I의 성능을 알아보기 위해 remote site의 실제 로봇을 local site에서 구현한 시뮬레이터의 가상 로봇과 동일한 모델로 구성하여 모의 실험을 한다. 모의 실험시 local site와 remote site는 그림 4.5와 같이 동일한 컴퓨터에 구현한다. 전체 구조는 두 site에 실제의 인터넷망과 반사기(Reflector)가 추가된 구조를 갖는다. 두 지역간의 물리적인 거리는 약 30km정도이다. 표 4.4은 모의 실험에서 사용한 매개변수 값들을 나타낸다.

그림 4.19과 그림 4.20은 인터넷 제어 구조 I을 이용하여 모의 실험한 결과를 나타낸다. 각각은 local site와 remote site에서의 이동 로봇 궤적을 나타낸다. 모의 실험은 장애물을 회피하여 환경의 왼쪽 상위 지점에서 오른쪽 하위 지점까지 도달하는 것을 목표로 한다. 모의 실험 결과에서 나타난 것처럼 local site의 가상 로봇이 불연속적으로 움직이는 경우가 빈번히 발생한다. 그림 4.21에 나타난 가상 로봇과 실제 로봇의 경로를 보면, 두 로봇의 궤적간에 많은 오차가 있음을 알 수 있다. 오차가 발생하는 이유는 샘플링 주기보다 큰 인터넷 시간 지연이 있은 후, 몇 개의 제어 입력 정보가 손실되기 때문이다. 그림 4.22의 모의 실험 결과에서 두 site간의 시간차가 증가한 후, 큰 기울기로 시간차가 줄어드는 영역에서 제어 입력 정보의 손실이 발생한다. 제어 입력의 손실이 발생하면 실제 로봇은 가상 로봇과 다른 움직임을 보이게 되고, 실제 로봇의 자세가 local site로 되먹임 되어 가상 로봇의 궤적에 불연속점이 생기게 된다. 모의 실험 중의 인터넷 시간 지연은 그림 4.23과 같다. 그림에서 실선은 local site에서 remote site로 제어 입력이 전송될 경우에 발생하는 인터넷 시간 지연이고, 점선은 remote site에서 local site로 자세가 되먹임될 경우에 발생하는 인터넷 시간 지연을 나타낸다.

인터넷 제어 구조 I의 경우, 인터넷 시간 지연이 샘플링 주기보다 작으면 경로 오차는 발생하지 않는다. 하지만 인터넷 시간 지연이 샘플링 주기보다 커지게 되면 가상 로봇과 실제 로봇간의 경로 오차가 발생하고, 가상 로봇의 궤적에는 불연속점이 발생하게 되어 사용자가 로봇을 제어하는데 큰 어려움을 느끼게 된다. 인터넷 시간 지연이 상당히 클 경우, 사용자가 원하는 목적지까지 로봇을 이동시키는 것은 거의 불가능하다.

Simulator	Sampling time	$T = 0.4$ sec
Posture Estimator	Gain	$K_e = 0.2$
Robot	Sampling time	$T_R = 50\ msec$
	Maximum velocity	$v_{\max} = 30\ cm/s$
	Maximum angular velocity	$\omega_{\max} = 6\ rad/s$
	Maximum acceleration	$a_{\max} = 50\ cm/s^2$
	Maximum angular acceleration	$\alpha_{\max} = 10\ rad/s^2$
	Size of Body	$W = 7.5\ cm$
	Length between two wheels	$L = 7\ cm$
	Radius of wheel	$R = 2\ cm$
Sensors	Number	$N_s = 5$
	Sensor site angle	$\theta_s = 25^\circ$
	Angle between two sensors	$\Delta\theta_s = 30^\circ$
	Maximum sensing distance	$D_{\max} = 40\ cm$
	Minimum sensing distance	$D_{\min} = 0\ cm$
	Sensor position	$R_s = 4.5\ cm$
Environment	Size X	$E_X = 312\ cm$
	Size Y	$E_Y = 153\ cm$

표 4.4: 모의 실험을 위한 매개변수 값 설정

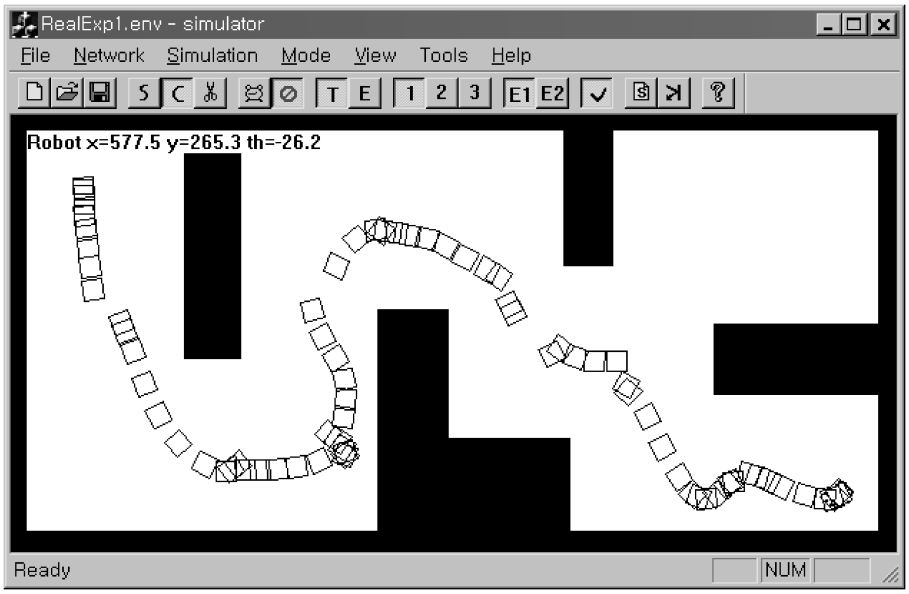


그림 4.19: Local site의 가상 로봇 이동 경로 (인터넷 제어 구조 I)

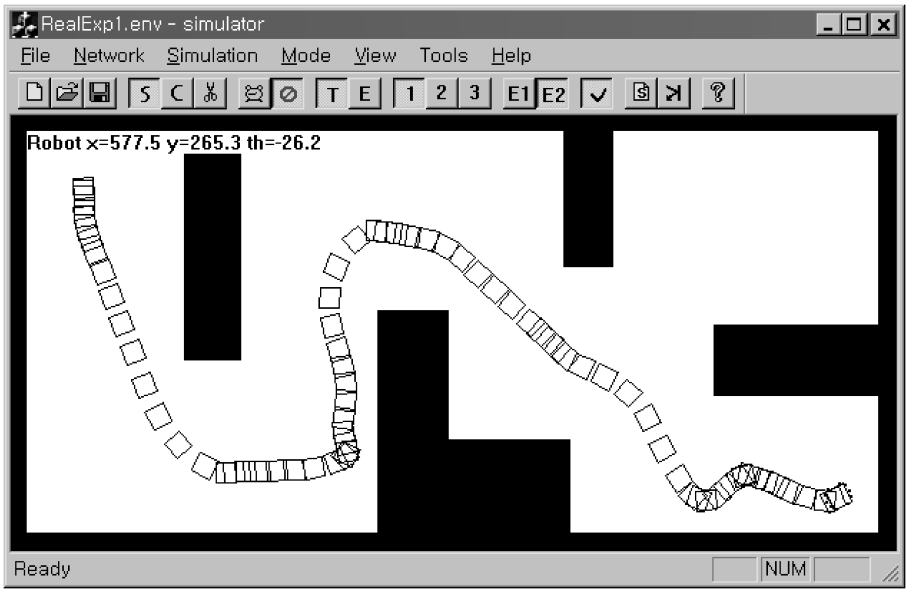


그림 4.20: Remote site의 실제 로봇 이동 경로 (인터넷 제어 구조 I)

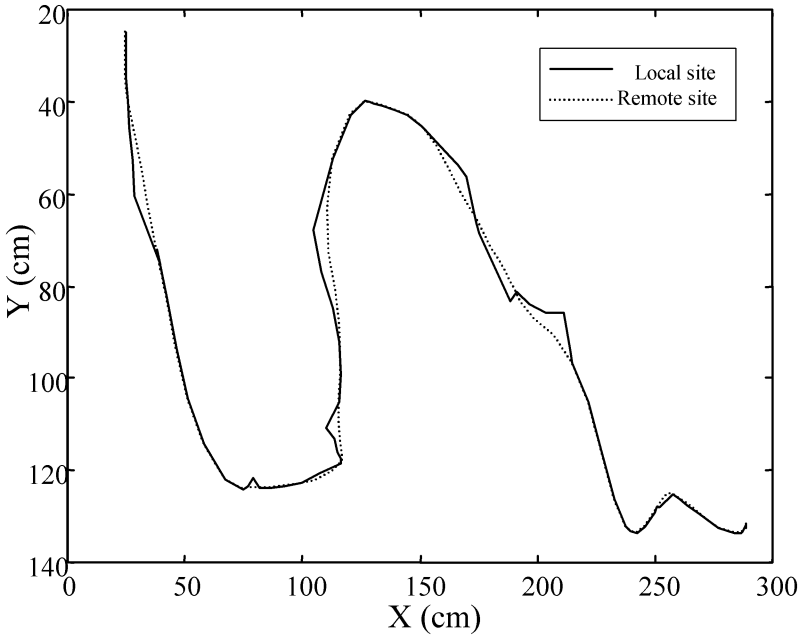


그림 4.21: 가상 로봇과 실제 로봇의 경로 오차 (인터넷 제어 구조 I)

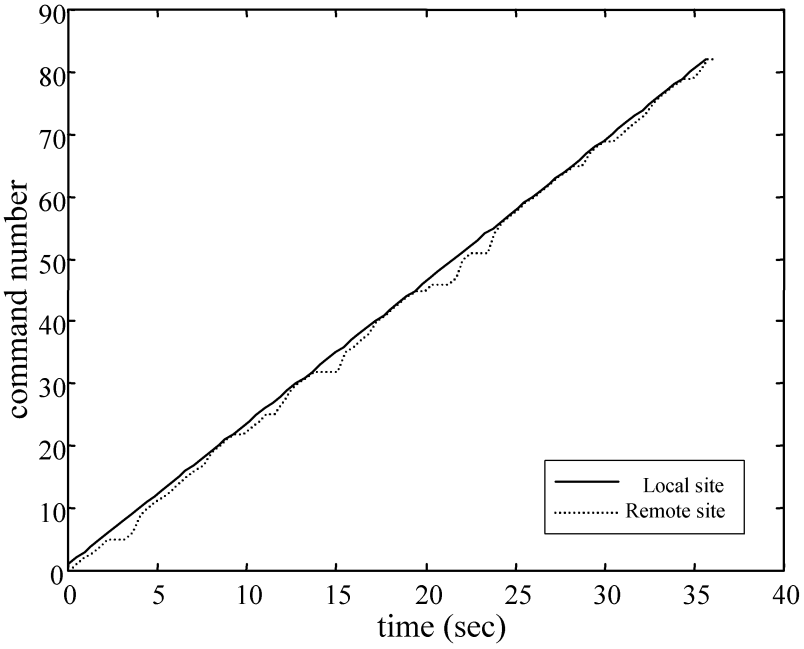


그림 4.22: 가상 로봇과 실제 로봇의 시간차 (인터넷 제어 구조 I)

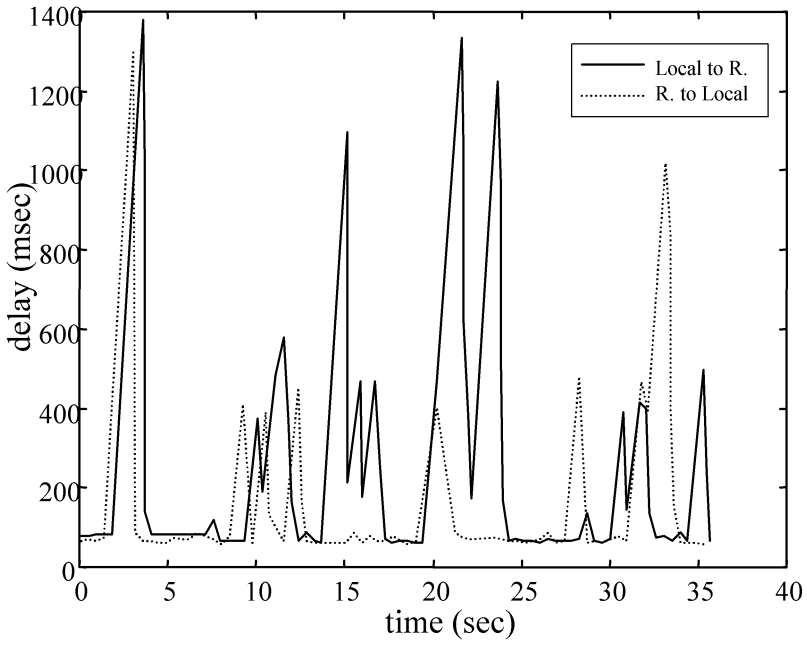


그림 4.23: 인터넷 제어 구조 I 모의 실험 중 인터넷 시간 지연

4.4.2 인터넷 제어 구조 II

인터넷 제어 구조 II는 인터넷 제어 구조 I의 시스템에 명령 처리 필터(Command Filter)가 추가된 구조를 갖는다. 인터넷 제어 구조 I은 인터넷 시간 지연이 샘플링 주기 T 보다 클 경우, 그림 4.16과 같이 제어 입력 정보의 손실이 발생한다. 이것은 몇 개의 제어 입력이 동시에 로봇으로 입력되기 때문이다. 인터넷 제어 구조 I의 문제점을 해결하기 위해 명령 처리 필터를 추가한 인터넷 제어 구조 II (그림 4.26)를 제안한다.

명령 처리 필터(Command Filter)는 인터넷망을 통해 전달되는 제어 입력을 로봇에 적합한 명령으로 변환한다. 오랜 인터넷 시간 지연이 있은 후, 정보의 손실을 막기 위해서 동시에 전달되는 여러 개의 제어 입력에 다시 샘플링 주기를 추가시킨다. 명령 처리 필터의 추가는 제어 입력의 손실을 제거하여 가상 로봇과 실제 로봇의 궤적 오차를 줄인다. 그림 4.24은 명령 처리 필터의 역할을 나타낸다.

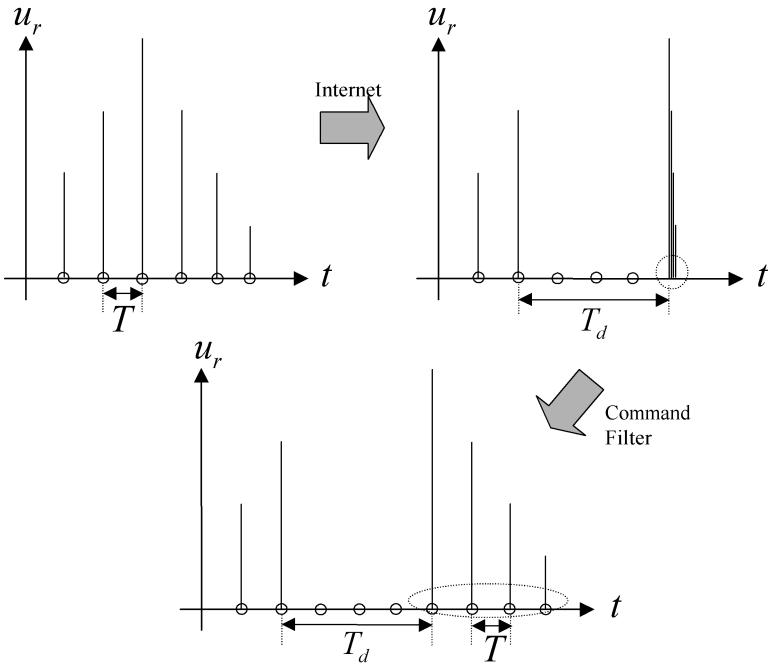


그림 4.24: Command Filter의 역할

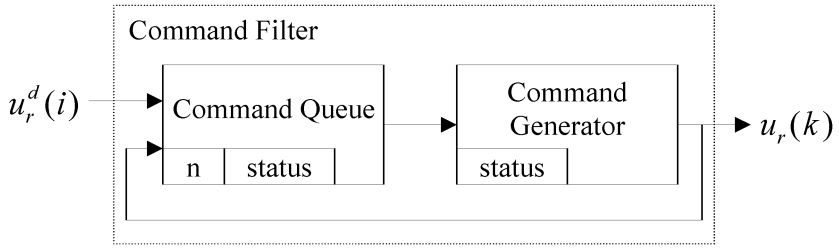


그림 4.25: Command Filter의 구조

명령 처리 필터(Command Filter)의 구조는 그림 4.25과 같이 명령 큐(Command Queue)와 명령 발생기(Command Generator)로 이루어진다. 명령 큐(Command Queue)와 명령 발생기(Command Generator), 그리고 명령 처리 필터(Command Filter)는 다음과 같이 DEVS(Discrete Event Variable System) Formalism으로 표현 가능하다. DEVS Formalism은 Atomic Model과 Coupled Model로 구성된다.

Atomic Model

$$AM = \langle X, Y, S, \delta_{ext}, \delta_{int}, \lambda, ta \rangle$$

X : *input events set*
Y : *output events set*
S : *sequential states set*
δ_{ext} : *external transition function*
($Q \times X \rightarrow S$, $Q = \{(s, e) | s \in S, 0 \leq e \leq ta(s)\}$)
δ_{int} : *internal transition function* ($S \rightarrow S$)
λ : *output function* ($S \rightarrow Y$)
ta : *time advance function* ($S \rightarrow \mathbf{R}^{+}_{0,\infty}$)

$$\therefore \; Queue = \langle X, Y, S, \delta_{ext}, \delta_{int}, \lambda, ta \rangle$$

$X = \{U_{in}, done\}$
$Y = \{U_{out}\}$
$S = \{(n, status) \mid n \in \{0,1,2,\cdots\},\ status \in \{Busy, Free\}\}$
δ_{ext} : $\delta_{ext}((n, \_), U_{in}) = (n+1, \_)$
$\delta_{ext}((\_, Busy), done) = (\_, Free)$
δ_{int} : $\delta_{int}((n \neq 0, Free)) = (n-1, Busy)$
λ : $\lambda((n \neq 0, Free)) = U_{out}$
ta : $ta((n \neq 0, Free)) = 0$
$otherwise \;\; ta = \infty$

$$\therefore\ Generator = \langle X, Y, S, \delta_{ext}, \delta_{int}, \lambda, ta \rangle$$

$$\begin{aligned}
X &= \{U_{in}\} \\
Y &= \{U_{out}\} \\
S &= \{status \mid status \in \{Busy, Free\}\} \\
\delta_{ext}: &\ \delta_{ext}(Free, U_{in}) = Busy \\
\delta_{int}: &\ \delta_{int}(Busy) = Free \\
\lambda: &\ \lambda(Busy) = U_{out} \\
ta: &\ ta(Busy) = T\ (sampling\ time) \\
&\ otherwise\ \ ta = \infty
\end{aligned}$$

Coupled Model

$$CM = \langle X, Y, M, EIC, EOC, IC, Select \rangle$$

$$\begin{aligned}
X: &\ input\ events\ set \\
Y: &\ output\ events\ set \\
M: &\ set\ of\ all\ component\ models\ in\ DEVS \\
EIC: &\ external\ input\ coupling\ relation \\
&\subseteq CM.IN \times M.IN \\
EOC: &\ external\ output\ coupling\ relation \\
&\subseteq M.OUT \times CM.OUT \\
IC: &\ internal\ coupling\ relation \\
&\subseteq M.OUT \times M.IN \\
Select: &\ tie-breaking\ selector\ \ (\ 2^M - \emptyset \rightarrow M\ )
\end{aligned}$$

$$\therefore\ Filter = \langle X, Y, M, EIC, EOC, IC, Select \rangle$$

$$\begin{aligned}
X &= \{U_{in}\} \\
Y &= \{U_{out}\} \\
M &= \{Queue,\ Generator\} \\
EIC &= \{(Filter.U_{in},\ Queue.U_{in})\} \\
EOC &= \{(Generator.U_{out},\ Filter.U_{out})\} \\
IC &= \{(Queue.U_{out},\ Generator.U_{in}), \\
&\quad\ (Generator.U_{out},\ Queue.U_{in})\} \\
Select: &\ \ Select(\{Queue,\ Generator\}) = Queue
\end{aligned}$$

명령 처리 필터(Command Filter)는 속도 제어 입력을 받아들여 명령 큐(Command Queue)에 저장한다. 명령 발생기(Command Generator)는 샘플링 주기 T 마다 명령 큐에 저장된 속도 제어 입력을 명령 처리 필터의 출력으로 내보낸다. 이때 명령 처리 필터의 입력과 출력은 각각 $U_{in} = u_r^d(i)$, $U_{out} = u_r(k)$의 관계임을 알 수 있다.

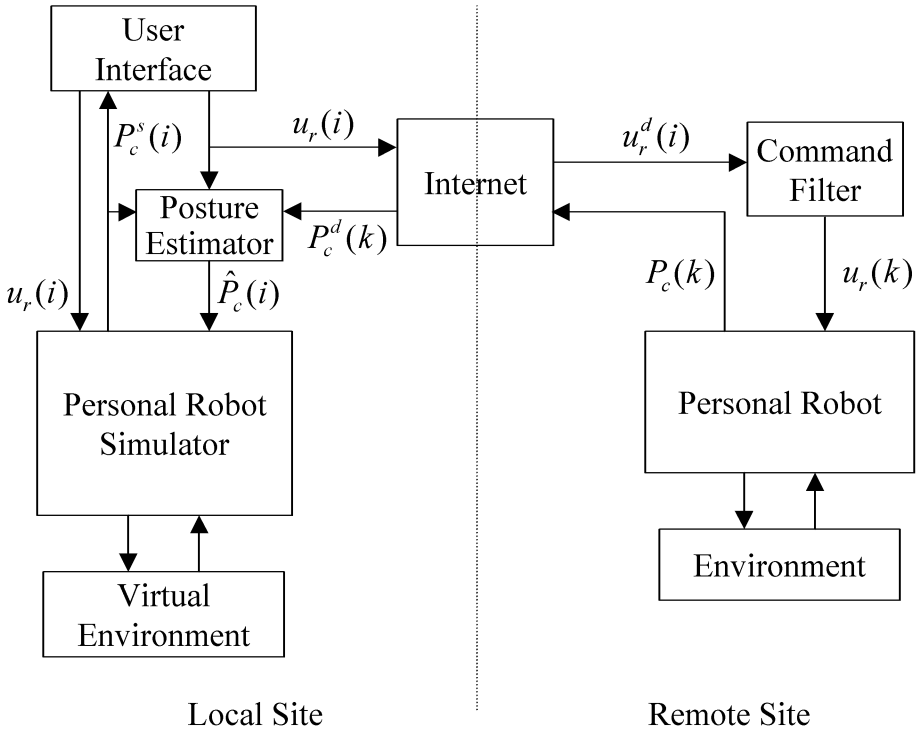


그림 4.26: 인터넷 제어 구조 II

$\boldsymbol{u}_r(i)$: 사용자에 의한 i 번째 제어 입력 $[v_r(i)\ \omega_r(i)]^T$

$\boldsymbol{u}_r^d(i)$: 인터넷망을 통과한 i 번째 제어 입력

$\boldsymbol{u}_r(k)$: Command Filter를 통과한 k 번째 제어 입력

$\boldsymbol{P}_c(k)$: 되먹임을 위한 k 번째 로봇 자세

$\boldsymbol{P}_c^d(k)$: 인터넷망을 통과한 k 번째 로봇 자세

$\widehat{\boldsymbol{P}_c}(i)$: 되먹임 값을 이용한 i 번째 로봇 자세 추정치

$\boldsymbol{P}_c^s(i)$: 가상 로봇의 i 번째 로봇 자세

인터넷 제어 구조 II의 모의 실험

인터넷 제어 구조 II를 위한 모의 실험 구조는 인터넷 제어 구조 I의 모의 실험 구조와 동일하다. 모의 실험에서 사용한 매개변수 값은 표 4.4과 같다.

그림 4.27와 그림 4.28의 모의 실험 결과에서는 인터넷 제어 구조 I의 모의 실험 결과에서 발생한 가상 로봇 궤적의 불연속점이 나타나지 않는다. 가상 로봇과 실제 로봇의 두 궤적은 오차가 상당히 작다는 것을 그림 4.29의 결과를 통해 확인할 수 있다. 이와 같은 실험 결과는 명령 처리 필터(Command Filter)의 역할로 설명이 가능하다. 명령 처리 필터(Command Filter)는 인터넷망을 통해 들어오는 제어 입력을 명령 큐(Command Queue)에 저장시킨 후, 샘플링 주기 간격으로 명령 발생기(Command Generator)를 통해 받아들이므로, 큰 인터넷 시간 지연에 의한 제어 입력 정보의 손실을 막을 수 있다. 모의 실험에서 실제 로봇 모델은 가상 로봇 모델과 동일하므로 제어 입력 정보의 손실이 발생하지 않으면, 실제 로봇은 가상 로봇과 같은 궤적을 형성하게 된다. 인터넷 제어 구조 II에서 경로 오차가 발생하는 원인은 크게 두 가지이다. 첫 번째 원인은 가상 로봇 모델과 실제 로봇간의 차이이고, 두 번째는 큰 인터넷 시간 지연으로 실제 로봇의 제어 입력 값이 0일 경우 로봇의 최대 감가속도로 속도가 줄어들고, 다음 제어 입력이 들어오면 다시 최대 가속도로 속도가 증가에 의한 오차이다. 모의 실험에서는 가상 로봇과 실제 로봇의 모델이 일치하기 때문에 경로 오차 원인은 후자의 경우만 해당된다. 후자에 의한 경로 오차는 로봇의 가속도가 클수록 줄어든다. 가속도가 무한대일 경우에는 가상 로봇과 실제 로봇의 궤적이 일치한다.

인터넷 제어 구조 II의 가상 로봇 궤적에 있어서 인터넷 제어 구조 I과 같은 불연속점이 존재하지 않는다는 것은 사용자가 가상 로봇을 쉽게 제어할 수 있다는 점에서 큰 의미를 갖는다. 하지만 인터넷 제어 구조 II는 그림 4.30에서 나타난 결과와 같이 가상 로봇과 실제 로봇간의 시간차가 줄어들지 않고, 점차 증가한다는 단점을 갖는다. 큰 인터넷 시간 지연이 발생하면, 그와 동일한 시간차가 추가된다. 그림 4.31의 결과에서 보면 10초에서 20초 사이에 큰 인터넷 시간 지연이 나타난 것을 알 수 있다. 이 때 발생한 가상 로봇과 실제 로봇간의 시간차(그림 4.30)는 계속 남아있게 된다.

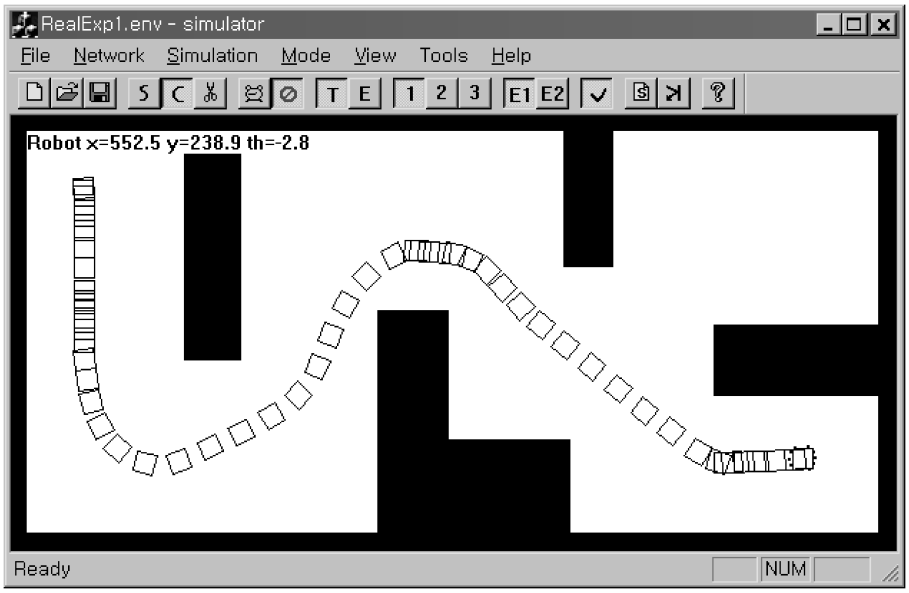


그림 4.27: Local site의 가상 로봇 이동 경로 (인터넷 제어 구조 II)

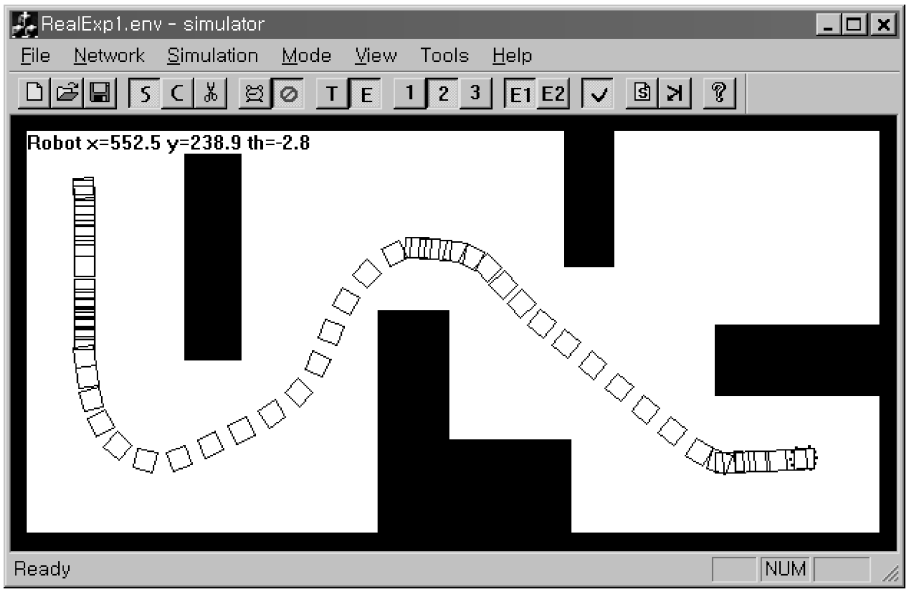


그림 4.28: Remote site의 실제 로봇 이동 경로 (인터넷 제어 구조 II)

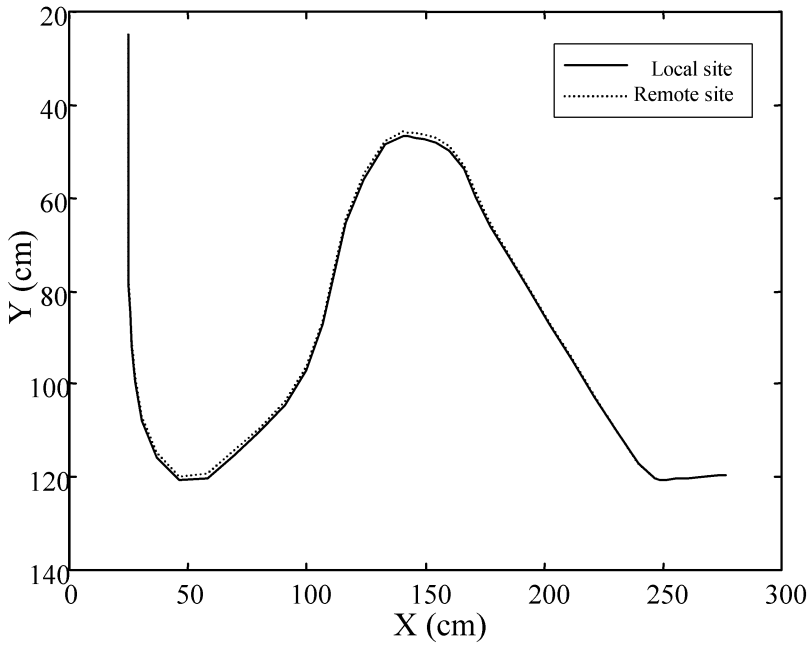


그림 4.29: 가상 로봇과 실제 로봇의 경로 오차 (인터넷 제어 구조 II)

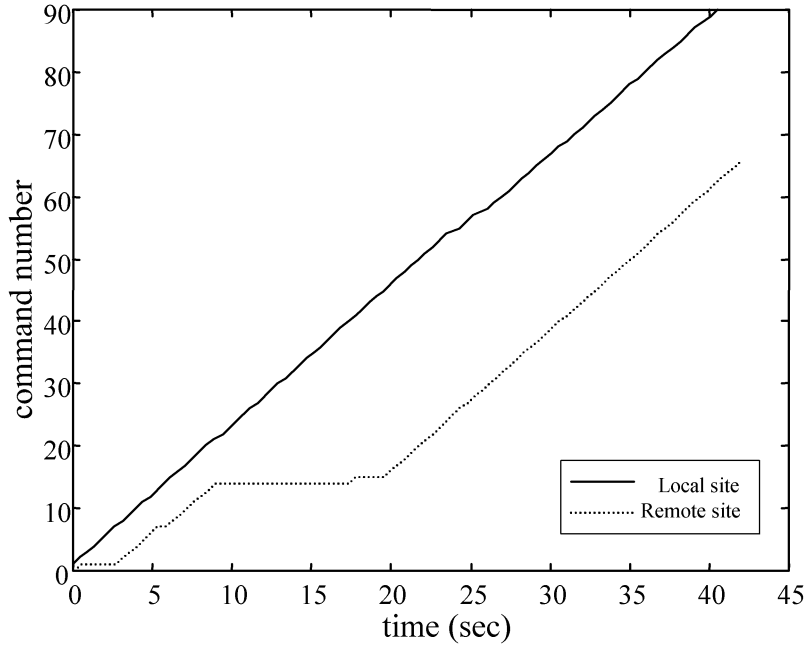


그림 4.30: 가상 로봇과 실제 로봇의 시간차 (인터넷 제어 구조 II)

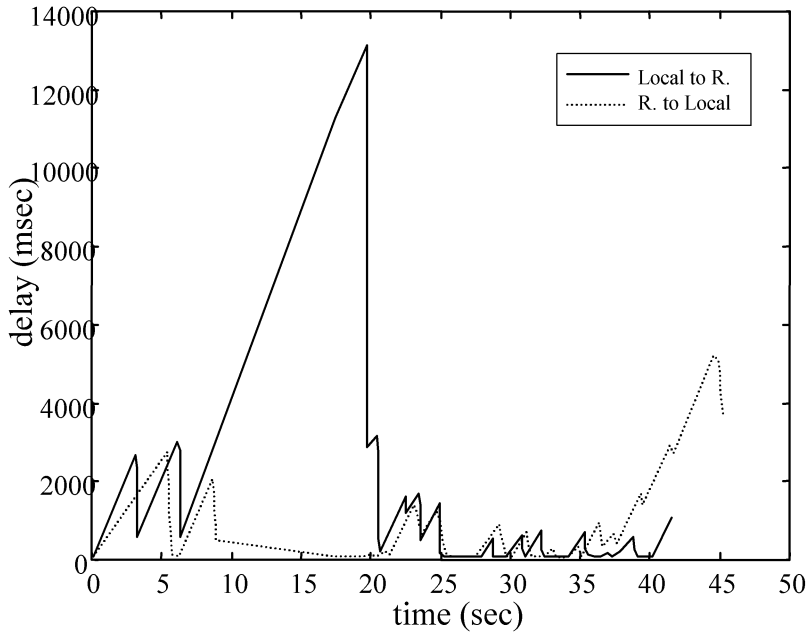


그림 4.31: 인터넷 제어 구조 II 모의 실험 중 인터넷 시간 지연

4.4.3 인터넷 제어 구조 III

인터넷 제어 구조 II는 인터넷 제어 구조 I의 제어 입력 정보가 손실되는 단점을 보완한다. 하지만 그림 4.24에 나타난 결과와 같이 인터넷 시간 지연은 점차 누적되어 가상 로봇과 실제 로봇의 움직임에 있어서 시간차가 커지게 된다. 인터넷 제어 구조 II는 전체 시스템의 구조 설계를 위해 설정한 문제중 세 번째 문항의 조건을 만족하지 못한다. 인터넷 시간 지연에 의한 두 로봇간의 시간차가 증가하는 문제를 해결하기 위해 그림 4.35와 같은 인터넷 제어 구조 III을 제안한다.

인터넷 제어 구조 III은 인터넷 제어 구조 II에 경로 생성기(Path Generator)와 경로 추종 제어기(Path-Following Controller)가 추가된 구조를 갖는다. 인터넷망을 통해 들어오는 제어 입력은 그대로 로봇의 입력으로 사용되지 않고, 먼저 경로 생성기(Path Generator)로 입력되어 시뮬레이터의 가상 로봇과 동일한 경로를 생성시켜 나간다. 생성되는 경로는 경로 추종 제어기(Path-Following Controller)로 입력되고, 경로 추종 제어기는 로봇의 제어 입력을 만들어 로봇이 경로를 따라가도록 제어한다.

인터넷 제어 구조 III은 인터넷망을 통과한 제어 입력과 실제 로봇의 입력을 분리시킴으로써 인터넷 시간 지연에 의한 시간차 문제를 해결한다. 명령 처리 필터(Command Filter)의 명령 발생기(Command Generator)에서 명령을 발생시키는 주기를 샘플링 주기 T 보다 작은 컴퓨터의 연산 처리 시간(processing time) T_p 로 바꿀 수 있다. 즉 제어 입력 정보의 손실을 막으면서 가상 로봇과 실제 로봇간의 시간차를 줄일 수 있다. 경로 생성기(Path Generator)에서는 시뮬레이터의 가상 로봇과 일치하는 모델을 사용하여 샘플링 주기마다 경로를 추가 생성하게 되므로, 생성된 경로는 가상 로봇의 경로와 일치한다. 그림 4.32는 인터넷 제어 구조 III에서 사용된 명령 처리 필터(Command Filter)의 동작을 나타낸다. 인터넷 시간 지연 T_d 이후, 명령 발생 시간이 샘플링 주기보다 작은 연산 처리 주기로 동작함을 의미한다. 이때 인터넷 제어 구조 II의 명령 처리 필터보다 줄어든 시간차는 다음과 같다.

$$\Delta t = NT - NT_p = N(T - T_p)$$

$$Generator = \langle X, Y, S, \delta_{ext}, \delta_{int}, \lambda, ta \rangle$$

$$\begin{aligned} X &= \{U_{in}\} \\ Y &= \{U_{out}\} \\ S &= \{status \mid status \in \{Busy, Free\}\} \\ \delta_{ext} : &\quad \delta_{ext}(Free, U_{in}) = Busy \\ \delta_{int} : &\quad \delta_{int}(Busy) = Free \\ \lambda : &\quad \lambda(Busy) = U_{out} \\ ta : &\quad ta(Busy) = T_p \ (processing\ time) \\ &\quad otherwise \quad ta = \infty \end{aligned}$$

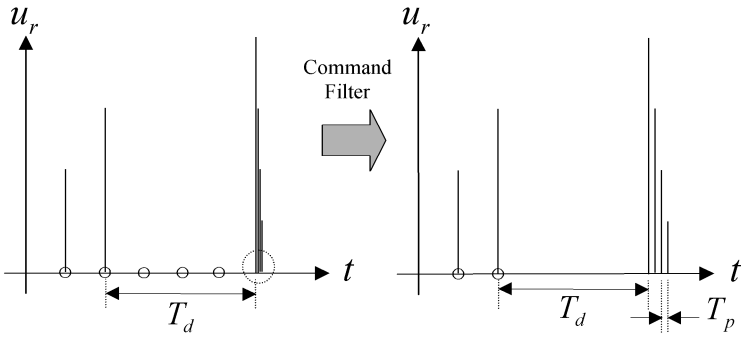


그림 4.32: 인터넷 제어 구조 III의 명령 처리 필터

인터넷 제어 구조 III의 경우, 경로 생성기(Path Generator)에서 생성된 경로가 입력되는 경로 추종 제어기(Path-Following Controller)는 전체 시스템의 성능에 많은 영향을 미친다. 생성된 경로와 가상 로봇의 경로 사이에 오차가 없는 경우, 경로 추종 제어기(Path-Following Controller)의 성능이 떨어진다면 실제 로봇은 가상 로봇과 많은 움직임의 오차를 보일 것이다. 경로 추종 제어기는 로봇이 기준 경로로부터 떨어져 있을 경우에는 빠른 수렴을 하도록 유도하고, 기준 경로 위에 로봇이 놓여 있을 경우에는 기준 경로를 그대로 따라가는 제어를 수행한다. 경로 추종 제어기에 대해서는 이미 많은 연구[21,22,23,24]가 이루어졌다. 본 논문에서는 단위 벡터장 항법 방식(Uni-Vector Field Navigation Method)[21]을 이용하여 경로 추종 제어기(Path-Following Controller)를 구현한다.

단위 벡터장 항법 방식(Uni-Vector Field Navigation Method)은 원하는 기준 경로를 중심으로 주변에 단위 벡터장을 형성시킨다. 형성된 단위 벡터는 모든 점에서 로봇의

이동 방향을 제시한다. 로봇은 자신이 위치한 점에 형성된 단위 벡터의 방향을 따라가면 최종적으로 원하는 경로로 수렴할 수 있다. 그림 4.33은 경로 추종 제어기에서 사용하는 단위 벡터장 항법 방식에 의해 형성된 단위 벡터장을 나타낸다.

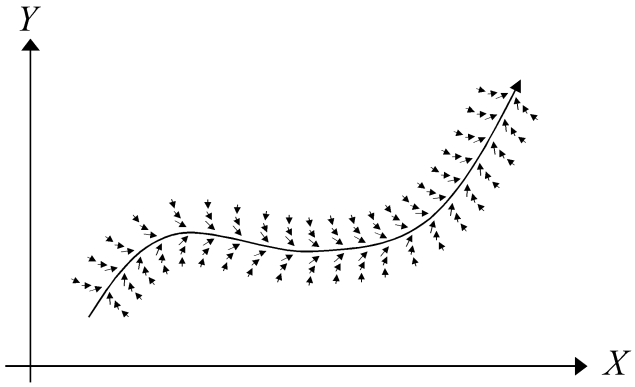


그림 4.33: 경로 추종 제어기의 단위 벡터장

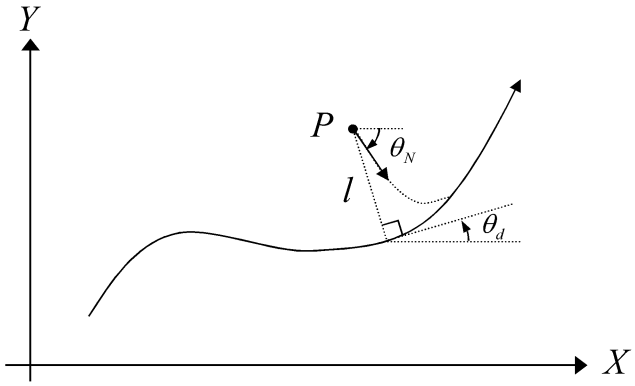


그림 4.34: 단위 벡터장 항법을 위한 변수 설정

임의의 점 P 에서의 단위 벡터가 이루는 각 θ_N 은 다음과 같이 정의할 수 있다.

$$\theta_N(l) = \theta_d - sgn(l)\tan^{-1}(g\,|l|^{1/c}), \quad g>0,\ c>1 \tag{4.8}$$

식(4.8)은 기준 경로로부터 떨어진 거리 l 이 클수록 단위 벡터가 경로와 이루는 각은 $\pm 90^\circ$ 에 가까와지고, 거리가 0에 가까와질수록 0° 로 점근함을 의미한다. 상수 c 값과 g 값은 형성되는 단위 벡터장의 모양을 결정한다. 상수 c 값이 작거나 g 값이 클수

록 기준 경로로 빠르게 수렴하는 벡터장이 형성된다. 두 상수의 역할은 같지만 수렴 속도의 변화량에는 차이가 있다. 두 값의 조합으로 원하는 속도로 수렴하는 벡터장을 형성할 수 있다.

경로 추종 제어기(Path-Following Controller)는 원하는 경로를 따라가기 위한 로봇의 제어 입력 $[v\ \omega]^T$ 값을 결정한다. θ_c 는 로봇의 자세 각도이고, θ_e 는 단위 벡터와 로봇 자세 각도의 오차이다. Δs는 경로의 끝점과 현재 로봇의 경로 거리 오차이다.

$$\begin{aligned} \therefore\ \omega &= K_\omega sgn(\theta_e)\sqrt{|\theta_e|} + (\cos\theta_c \frac{\partial\theta_N}{\partial x} + \sin\theta_c \frac{\partial\theta_N}{\partial y})\ v \\ v &= K_v \Delta s \end{aligned} \tag{4.9}$$

식(4.9)의 제어 입력을 이용하면 θ_e는 0으로 수렴하게 된다. θ_e의 수렴은 Lyapunov 정리를 이용하여 증명할 수 있다. 식(4.10)과 같이 Lyapunov 함수를 선택한다.

$$V = \frac{1}{2}\theta_e^2 \tag{4.10}$$

$$\begin{aligned} \theta_e &= \theta_N - \theta_c \\ \dot{\theta}_e &= \dot{\theta}_N - \dot{\theta}_c \\ &= (\frac{\partial\theta_N}{\partial x}\frac{dx}{dt} + \frac{\partial\theta_N}{\partial y}\frac{dy}{dt}) - \omega \\ &= (\frac{\partial\theta_N}{\partial x}\ v\cos\theta_c + \frac{\partial\theta_N}{\partial y}\ v\sin\theta_c) - \omega \\ &= -K_\omega sgn(\theta_e)\sqrt{|\theta_e|} \end{aligned}$$

$$\therefore\ \dot{V} = \theta_e\dot{\theta}_e = -K_\omega|\theta_e| \le 0 \tag{4.11}$$

Lyapunov 함수의 시간에 대한 미분은 식(4.11)과 같고, 그 값은 항상 0보다 작거나 같다. 이 때 미분값이 0과 같은 경우는 θ_e의 값이 0일 경우이다. 따라서 θ_e는 0으로 수렴하게 된다. $\theta_e = 0$으로 수렴하는 시간은 다음과 같이 구할 수 있다.

$$\dot{\theta}_e = -K_\omega sgn(\theta_e)\sqrt{|\theta_e|}$$

$$\theta_e(t) = \begin{cases} (\sqrt{\theta_e(0)} - \frac{K_\omega}{2}t)^2 & , \ \theta_e \geq 0 \\ -(\sqrt{-\theta_e(0)} - \frac{K_\omega}{2}t)^2 & , \ \theta_e < 0 \end{cases}$$

$$\therefore \ t|_{\theta_e=0} = \frac{2\sqrt{|\theta_e(0)|}}{K_\omega} \leq \frac{\sqrt{2\pi}}{K_\omega} \quad (\because |\theta_e(0)| \leq \frac{\pi}{2})$$

식(4.9)와 같은 제어 입력을 이용하면 일정한 시간 내에 형성된 벡터장으로 수렴함을 확인할 수 있다. 로봇은 벡터장으로 수렴한 이후 계속 형성된 벡터장를 따라가게 된다. 로봇이 기준 경로로 수렴하기 위해서는 형성된 벡터장이 기준 경로로 수렴해야 한다.

다음은 기준 경로가 $y=0$ ($\theta_d=0$)이고, 벡터장으로 수렴한 로봇의 초기 위치가 (x_0, y_0), $y_0>0$ 인 경우에 대해 로봇이 기준 경로로 수렴함을 증명한다.

$$\theta_N = \theta_d - \tan^{-1}(gl^{1/c}) = -\tan^{-1}(gy^{1/c})$$

$$\frac{dy}{dx} = -gy^{1/c}$$

$$y = \left\{ -\frac{g(c-1)}{c}(x-x_0) + y_0^{\frac{c-1}{c}} \right\}^{\frac{c}{c-1}}$$

$$\therefore \ x|_{y=0} = x_0 + \frac{c}{g(c-1)} y_0^{\frac{c-1}{c}}$$

$$\Delta x = x|_{y=0} - x_0 = \frac{c}{g(c-1)} y_0^{\frac{c-1}{c}} \qquad (4.12)$$

형성된 벡터장은 식(4.12)와 같이 일정한 구간 내에서 원하는 경로로 수렴하고 있음을 확인할 수 있다. 본 논문에서는 경로 추종 제어기 구현시 $g=1$, $c=2$ 로 설정하며, 형성되는 벡터장과 경로 $y=0$ (초기 위치 (x_0, y_0))에 대한 수렴 거리는 다음과 같다.

$$\theta_N(l) = \theta_d - sgn(l)\tan^{-1}(\sqrt{|l|})$$

$$\Delta x = 2\sqrt{y_0} \qquad (4.13)$$

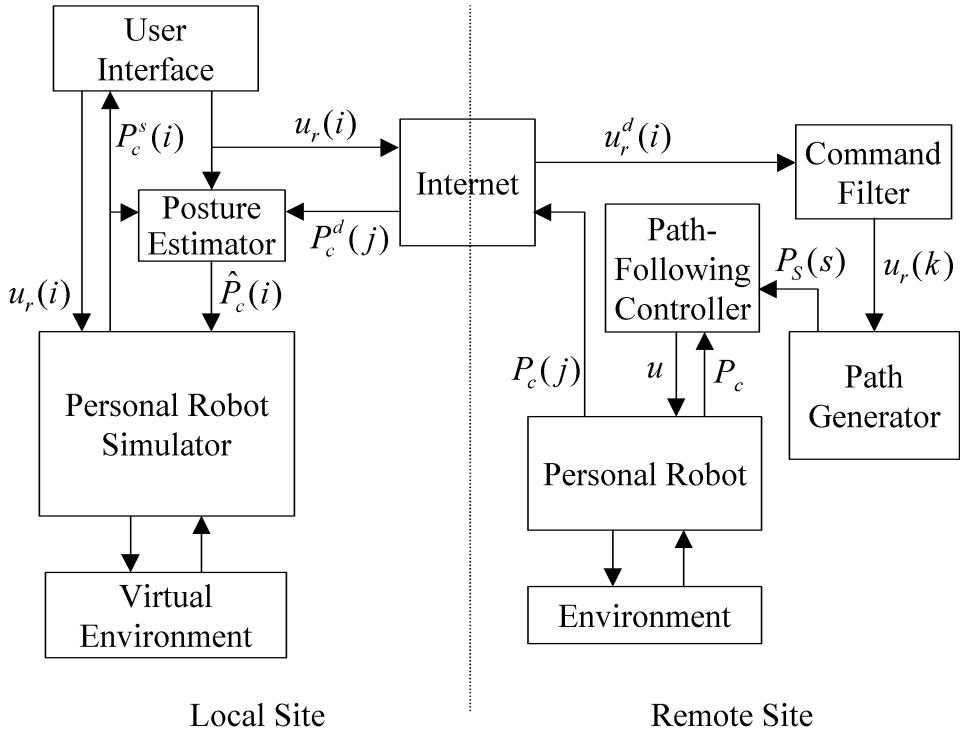


그림 4.35: 인터넷 제어 구조 III

$\boldsymbol{u}_r(i)$: 사용자에 의한 i 번째 제어 입력 $[v_r(i)\ \omega_r(i)]^{\mathrm{T}}$

$\boldsymbol{u}_r^d(i)$: 인터넷망을 통과한 i 번째 제어 입력

$\boldsymbol{u}_r(k)$: Command Filter를 통과한 k 번째 제어 입력

$\boldsymbol{P}_S(s)$: 가상 로봇의 움직임 궤적

$\boldsymbol{u}$: 경로 추적 제어기에 의한 로봇의 제어 입력

$\boldsymbol{P}_c$: 실제 로봇의 현재 자세

$\boldsymbol{P}_c(j)$: 되먹임을 위한 j 번째 로봇 자세

$\boldsymbol{P}_c^d(j)$: 인터넷망을 통과한 j 번째 로봇 자세

$\widehat{\boldsymbol{P}_c}(i)$: 되먹임 값을 이용한 i 번째 로봇 자세 추정치

$\boldsymbol{P}_c^s(i)$: 가상 로봇의 i 번째 로봇 자세

인터넷 제어 구조 III의 모의 실험

인터넷 제어 구조 III의 모의 실험 결과는 그림 4.36, 그림 4.37와 같다. 인터넷 제어 구조 II의 모의 실험 결과와 마찬가지로 가상 로봇 궤적의 불연속점은 존재하지 않는다. 이미 언급한 바와 같이 가상 로봇의 궤적에 불연속점이 존재하게될 경우, 사용자는 시뮬레이터를 통해 가상 로봇을 제어하는데 큰 어려움을 느끼게 된다.

인터넷 제어 구조 III에서 가상 로봇과 실제 로봇 궤적의 오차는 경로 추종 제어기(Path-Following Controller)의 성능에 따라 결정된다. 명령 처리 필터(Command Filter)와 경로 생성기(Path Generator)를 이용하여 가상 로봇의 궤적과 동일한 경로를 얻을 수 있다. 생성된 경로를 실제 로봇이 정확히 따라갈 수 있는 제어가 이루어질 때 오차가 발생하지 않는다. 본 논문에서 구현한 경로 추종 제어기는 이론상으로 오차를 없앨 수 있지만, 모의 실험에서는 로봇의 샘플링 주기($T_R = 50\ msec$)가 크기 때문에 약간의 오차가 발생하게 된다. 그림 4.38의 결과를 보면 가상 로봇의 궤적과 실제 로봇의 궤적은 거의 일치함을 확인할 수 있다.

인터넷 제어 구조 II는 궤적의 오차는 줄일 수 있지만 가상 로봇과 실제 로봇의 시간차는 줄일 수 없었다. 인터넷 제어 구조 III은 이와 같은 단점을 보완한다. 그림 4.39의 결과는 큰 인터넷 시간 지연의 영향에 의한 시간차가 점차 증가하지 않는다는 것을 보여준다. 시간차가 줄어든다는 것은 생성된 경로를 가상 로봇의 속도보다 더 빠르게 따라간다는 것을 의미한다. 인터넷 시간 지연이 작을 경우에는 가상 로봇의 속도와 실제 로봇의 속도는 유사하다. 하지만 인터넷 시간 지연이 커지게 되면 실제 로봇의 속도는 증가하게 되어 두 로봇간의 시간차는 줄어들게 된다.

그림 4.40은 인터넷 제어 구조 III의 모의 실험 중 발생한 인터넷 시간 지연을 나타낸다. 비교적 큰 시간 지연이 발생했음을 알 수 있다. 큰 시간 지연이 있음에도 불구하고 가상 로봇과 실제 로봇 경로 오차가 작고, 두 로봇간의 시간차는 감소하는 경향을 보였다. 인터넷 제어 구조 III은 4.1절에서 설정한 문제를 모두 해결함을 확인할 수 있다.

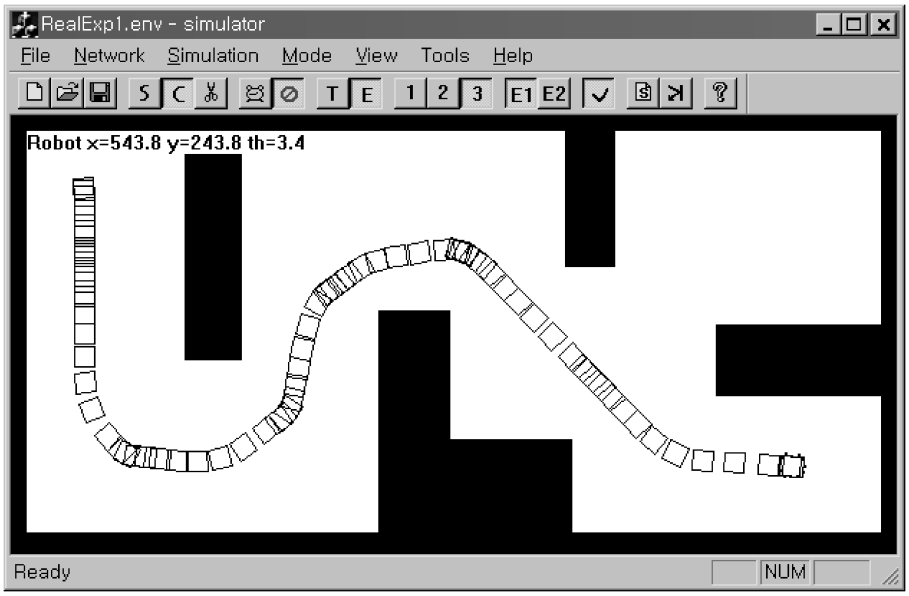


그림 4.36: Local site의 가상 로봇 이동 경로 (인터넷 제어 구조 III)

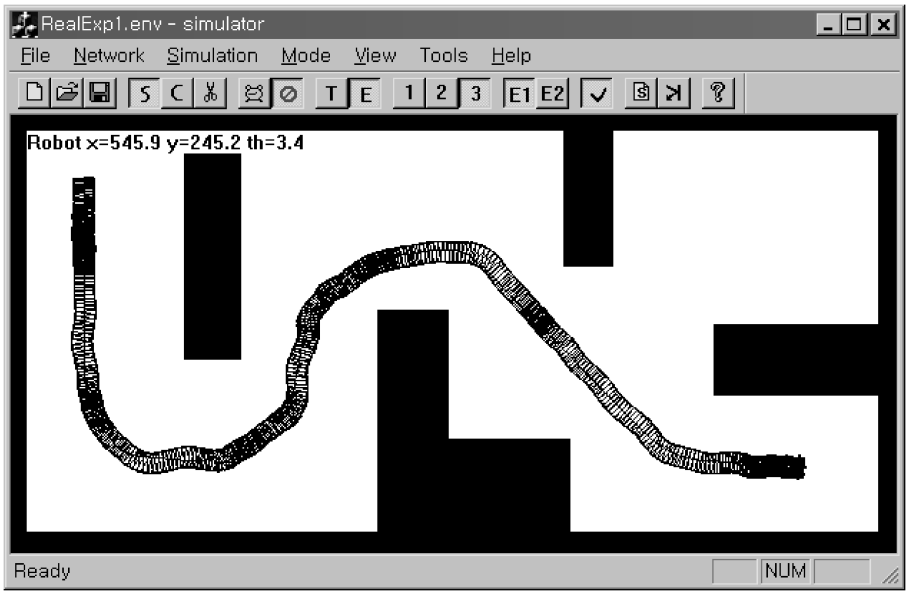


그림 4.37: Remote site의 실제 로봇 이동 경로 (인터넷 제어 구조 III)

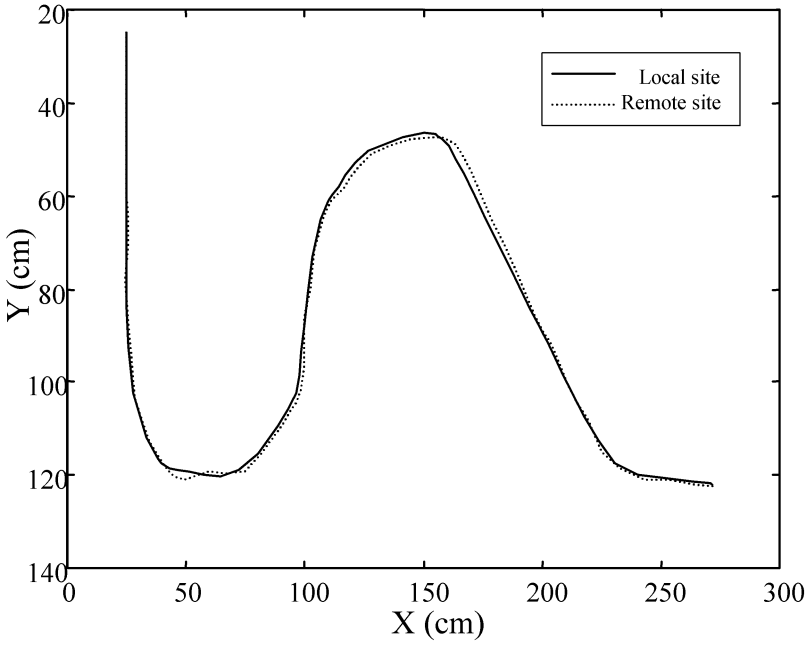


그림 4.38: 가상 로봇과 실제 로봇의 경로 오차 (인터넷 제어 구조 III)

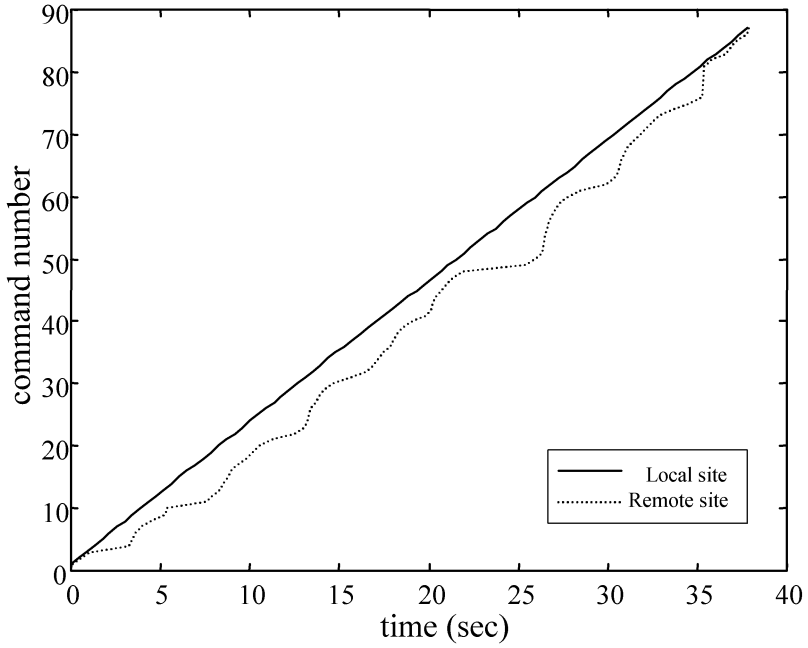


그림 4.39: 가상 로봇과 실제 로봇의 시간차 (인터넷 제어 구조 III)

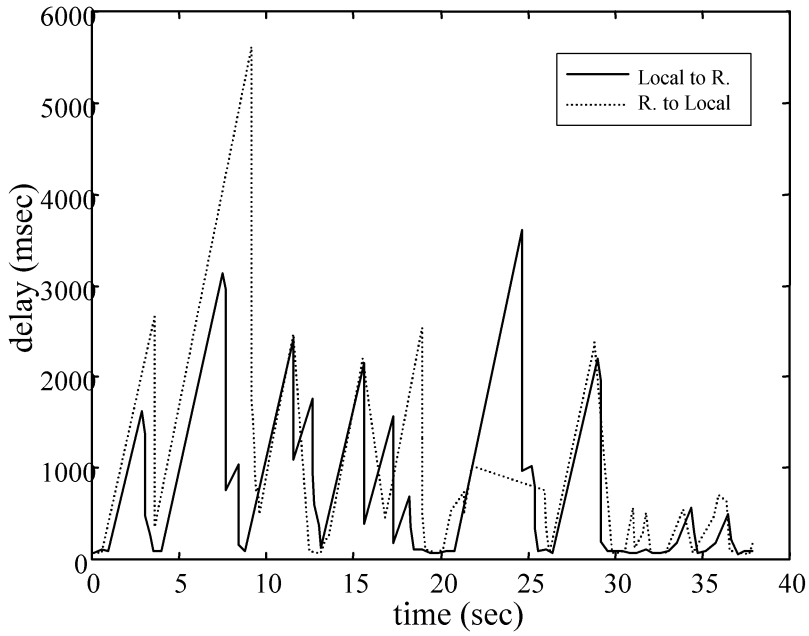


그림 4.40: 인터넷 제어 구조 Ⅲ 모의 실험 중 인터넷 시간 지연

4.5 환경 변화를 고려한 전체 시스템의 구조

그림 4.41의 구조는 실제 로봇의 환경과 가상 환경의 차이에서 오는 문제점을 해결하기 위한 수단으로 제안된다. 가상 환경은 이미 알고 있는 실제 환경의 정보를 바탕으로 구현된다. 사용자는 가상 환경을 보고 로봇을 제어하게 되므로, 가상 환경을 실제 환경으로 간주하게 된다. 이때 실제 환경의 변화로 가상 환경과 차이를 보이게 되면, 사용자에 의한 가상 로봇의 움직임과 실제 로봇의 움직임에 큰 차이가 발생할 확률이 높아진다. 따라서 실제 환경의 변화가 있을 경우, 가상 환경과 실제 환경간의 차이를 줄이는 방법에 대한 연구[25]가 요구된다.

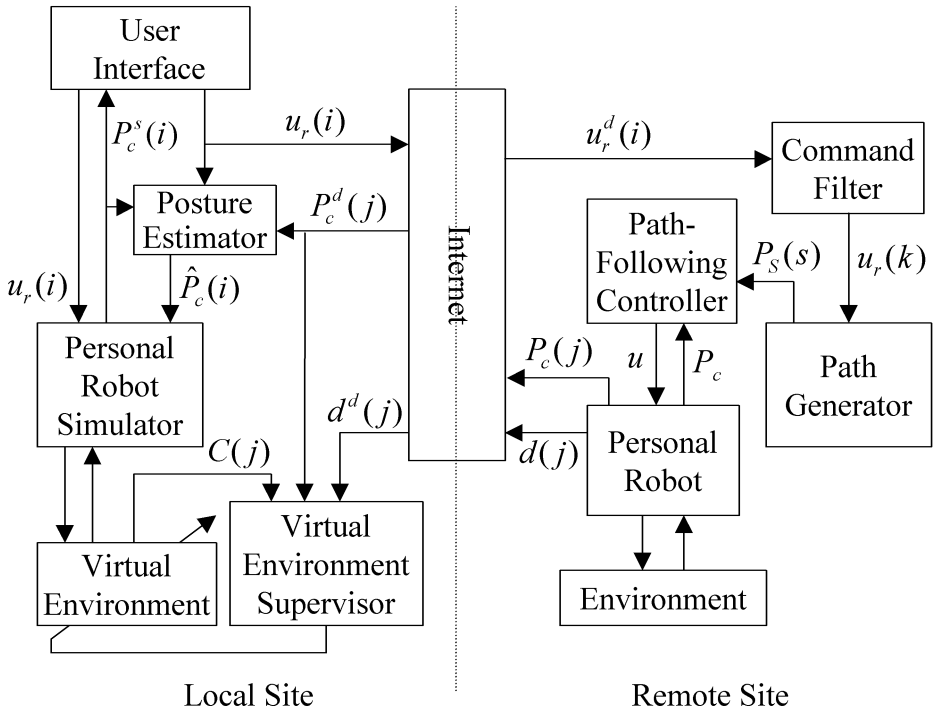


그림 4.41: 환경 변화를 고려한 전체 시스템의 구조

$\boldsymbol{d}(j)$: 되먹임을 위한 j 번째 센서 정보 $[\ d_0(j)\ \ d_1(j)\ \ \cdots\ \ d_{N_s}(j)\ ]^T$

$\boldsymbol{C}(j)$: 가상 환경의 j 번째 셀 정보

본 절에서 제안된 구조는 실제 로봇으로부터 센서 정보를 받아들이는 가상 환경 관리자(Virtual Environment Supervisor)가 추가된다. 가상 환경 관리자는 크게 세 가지의 기능을 수행한다. 첫 번째는 받아들인 센서 정보를 장애물로 인식할 것인지를 결정하는 기능이고, 두 번째는 센서 정보에 의한 장애물이 가상 환경에 이미 존재하는 장애물인지를 판별하는 기능이다. 마지막으로 세 번째는 센서 정보에 의한 장애물을 가상 환경의 장애물로 추가시키는 기능이다.

4.5.1 센서 정보에 의한 장애물의 인식

로봇의 센서 정보에 나타난 값을 그대로 장애물의 정보로 사용할 경우에 몇 가지 문제가 있다. 로봇의 전방에서 멀어지는 물체나 로봇 센서의 잡음 정보 등을 모두 장애물로 판별할 가능성이 크다. 가상 환경에 장애물이 추가되면 가상 로봇은 그 위치를 지나갈 수 없게 되고, 사용자도 추가된 장애물을 실제 장애물로 판단하는 상황이 발생한다. 따라서 센서에 나타난 정보를 장애물로 인식하기 위한 기준이 필요하다.

장애물은 로봇의 주행을 방해하는 물체를 의미한다. 주행에 경로 위에 있는 고정된 물체나 로봇으로 점차 접근하는 물체 등은 장애물에 속한다. 로봇과 멀리 떨어져 있거나 주행과는 무관한 작은 물체, 또는 로봇으로부터 점차 멀어지는 물체 등은 장애물로 인식할 필요가 없다. 본 논문에서는 로봇의 속도를 0으로 보았을 경우에 로봇으로 접근하는 물체의 상대 속도를 이용하여 장애물의 여부를 결정한다. 센서의 동작 샘플링 주기는 T_s, m 번째 센서의 방향으로 접근하는 물체의 속도는 v_{O_m}, 최소 속도는 $v_{O_{TH}}$, m 번째 센서의 j 번째 측정값은 $d_m(j)$ 이다. 센서 정보를 이용하여 장애물의 존재 여부를 판단하는 기준은 식(4.14)와 같다.

$$\Delta d_m = d_m(j-1) - d_m(j)$$

$$v_{O_m} = \frac{\Delta d_m}{T_s}$$

$$\max_m \; v_{O_m} \geq v_{O_{TH}} \tag{4.14}$$

장애물의 여부를 판단하는 작업은 remote site에서 수행하는 것이 더 유리하다. Remote site에서 장애물의 여부를 판단하여 인식된 장애물이 없을 경우에는 센서 정보를 되먹임시키지 않아도 된다. 즉 두 site 간의 정보 전송량을 줄일 수 있다는 장점을 갖는다.

4.5.2 인식된 장애물의 판별

식(4.14)에 의해 인식된 장애물을 가상 환경에 적용하기 위해서는 인식된 장애물이 이미 가상 환경에 존재하는 장애물인지의 여부를 판별해야 한다. 이와 같은 판별을 수행하지 않을 경우에는 센서 오차에 의해 이미 존재하는 장애물의 형태를 왜곡시킬 가능성이 높다. 센서의 오차가 없다고 가정하면 이미 가상 환경에 존재하는 장애물과의 판별식은 식(4.15)와 같이 정의할 수 있다. 이때 $d_m^s(j)$는 가상 환경에서 동작하는 가상 로봇의 m 번째 센서의 j 번째 측정값이다.

$$| d_m^s(j) - d_m(j) | > 0 \tag{4.15}$$

식(4.15)의 조건을 만족할 경우, 센서 정보에 의해 인식된 장애물은 가상 환경에 존재하지 않는 새로운 장애물로 판단한다. 조건을 만족하지 않는 경우, 센서 정보에 의해 인식된 장애물은 가상 환경에 이미 존재하는 장애물과 동일하다고 판단한다. 새로운 장애물의 위치는 다시 되먹임 받은 센서의 정보와 로봇의 자세 정보를 이용하여 알아낸다.

4.5.3 가상 환경에 대한 새로운 장애물의 적용

센서 정보에 의해 인식된 장애물이 가상 환경에 존재하지 않는 새로운 장애물로 판별된 경우에는 새로운 장애물에 대한 정보를 가상 환경에 포함시킨다. 장애물에 대한 센서 정보와 실제 로봇의 자세 정보를 이용하여 장애물의 좌표를 구할 수 있다. 가상 환경은 셀(cell) 단위로 모델링되어 있기 때문에, 장애물의 좌표를 이용하면 해당 셀을 찾을 수 있다. 셀 하나의 크기는 $1cm \times 1cm$, 센서의 측정값은 cm 단위를 갖는다고 가정하면 m 번째 센서에 대한 새로운 장애물의 설정은 다음과 같다. 이때 i 와 j 는 정수 값을 갖는다.

$$\begin{aligned} i &= x_c + d_m \cos(\theta_c + \Delta\theta_s m) \\ j &= y_c + d_m \sin(\theta_c + \Delta\theta_s m) \end{aligned} \qquad (4.16)$$

$$C_{ij} = 1$$

가상 환경에 새로 형성된 장애물은 실제 환경에서 다시 사라질 확률을 갖는다. 또한 센서의 잡음으로 인해 장애물이 형성된 경우도 발생할 수 있다. 위와 같이 추가된 장애물을 가상 환경에 영구적으로 포함시키는 경우에는 문제가 발생할 수 있다. 실제 환경에 없는 장애물을 사용자와 가상 로봇은 가상 환경에 장애물이 있는 것으로 인식하게 된다. 이러한 문제는 센서 정보를 이용하여 장애물을 제거하거나, 또는 가상 환경의 추가된 셀(cell)에 생존 기간(life-time)을 부여함으로써 해결이 가능하다. 셀의 생존 기간(life-time)은 주변 셀의 값에 의해 결정된다. 주변의 셀이 장애물로 인식되어 있을 경우에는 생존 기간이 길어지고, 주변의 셀에 장애물이 감지되지 않을 경우에는 생존 기간이 짧아진다. 이것은 센서의 잡음에 의해 형성된 셀의 정보를 빠른 시간 내에 제거함을 의미한다. 또한 크기가 작은 장애물보다는 큰 장애물의 생존 기간을 길게 설정한다는 것을 뜻한다. 생성된 셀(cell)의 생존 기간은 시간에 따른 함수로 표현이 가능하다. 셀(cell)의 생명(life)을 L 이라고 정의한다. 셀이 생성된 시점을 t_0 라 하면 시간에 따른 셀의 생명은 다음과 같이 표현할 수 있다.

$$L_{ij}(t - t_0) = e^{-A_{ij}(t - t_0)} \qquad (4.17)$$

$C_{i-1,j-1}$	$C_{i,j-1}$	$C_{i+1,j-1}$
$C_{i-1,j}$	$C_{i,j}$	$C_{i+1,j}$
$C_{i-1,j+1}$	$C_{i,j+1}$	$C_{i+1,j+1}$

그림 4.42: 인접한 셀(cell)의 정의

지수 함수의 형태를 결정짓는 매개변수 A_{ij}는 주변 셀(cell)의 값과 설계 매개변수로 다시 설정할 수 있다. 인접한 셀의 값을 모두 합한 값을 C_{ij}^{S} 로 정의한다. γ 는 설계 매개변수이고, L_{TH}은 셀의 생존 여부를 결정짓는 임계값이다.

$$\therefore\ C_{ij}(t-t_0) = sgn(\max(L_{ij}(t-t_0), L_{TH}) - L_{TH}) \tag{4.18}$$

where

$$sgn(0) = 0$$

$$L_{ij}(t-t_0) = e^{-\frac{(t-t_0)}{\gamma C_{ij}^{S}}}$$

$$C_{ij}^{S} = \sum_{p=i-1}^{i+1}\sum_{q=j-1}^{j+1} C_{pq} - C_{ij}$$

식(4.18)을 이용하여 셀의 생존 기간을 결정함으로써 센서 잡음에 의한 정보나 변화 가능성이 높은 작은 장애물 등에 대한 정보를 짧은 시간 내에 제거하는 것이 가능하다.

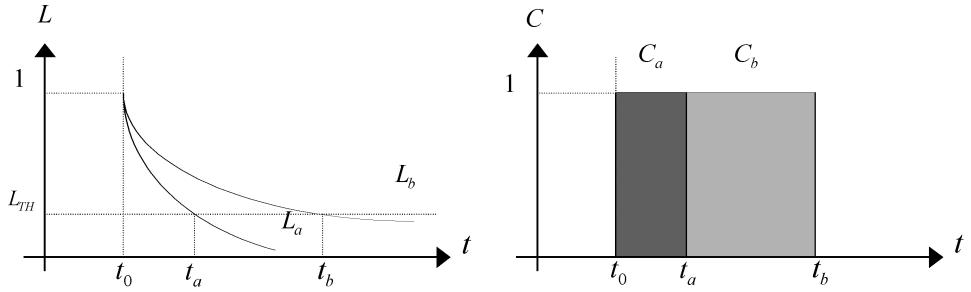


그림 4.43: 셀(cell)의 생존 기간 비교

4.5.4 모의 실험

가상 환경과 실제 환경에 차이가 생길 경우, 가상 환경을 실제 환경과 같게 만들어주기 위해 제안된 구조는 그림 4.41과 같고, 모의 실험 결과는 그림 4.44, 그림 4.45와 같다. 모의 실험에서 사용한 매개변수 값은 센서 관련 정보만을 제외하고 모두 표 4.4에 나타난 값과 동일하다. 환경 정보를 보다 자세히 얻기 위해 본 절의 모의 실험에서는 센서의 사양을 표 4.5와 같이 변경한다.

그림 4.44에서와 같이 초기의 가상 환경은 실제 환경과는 다른 정보를 갖는다. 실제 환경의 중앙 부근에 위치한 장애물이 가상 환경에는 존재하지 않는다. 가상 환경은 실제 로봇의 자세 정보와 센서 정보를 받아 초기에는 존재하지 않았던 중앙 부근의 장애물의 윤곽을 점차 포함시킨다. 모의 실험 결과에서 초기에는 가상 환경에 존재하지 않았던 장애물이 실제 장애물과 유사한 위치와 형태로 형성되어 있음을 확인할 수 있다.

본 절에서 제안한 구조는 센서 정보를 전달받기 이전에는 정확한 장애물의 정보를 얻을 수 없기 때문에, 가상 로봇이 이미 지나친 지역에 장애물이 나타날 수 있다. 이러한 경우, 가상 로봇의 움직임에 불연속점이 존재하게 된다. 없었던 장애물의 정보를 받아 가상 환경에 적용함과 동시에 가상 로봇은 현 위치에서 장애물 이전의 위치로 이동하게 된다. 이와 같은 상황이 빈번히 발생할 경우, 사용자가 가상 로봇을 제어하는데 어려움을 느끼게 된다. 하지만 이미 알고 있는 환경 정보 이외에 수시로 바뀌는 환경에 대한 정보를 사용자가 정확히 알아낼 수 있다는 점에서 제안된 구조는 큰 의미를 갖는다.

Sensors	Number	$N_s = 18$
	Sensor site angle	$\theta_s = 10^\circ$
	Angle between two sensors	$\Delta\theta_s = 10^\circ$
	Maximum sensing distance	$D_{\max} = 50\ cm$
	Minimum sensing distance	$D_{\min} = 0\ cm$
	Sensor position	$R_s = 4.5\ cm$

표 4.5: 모의 실험을 위한 센서 관련 매개변수

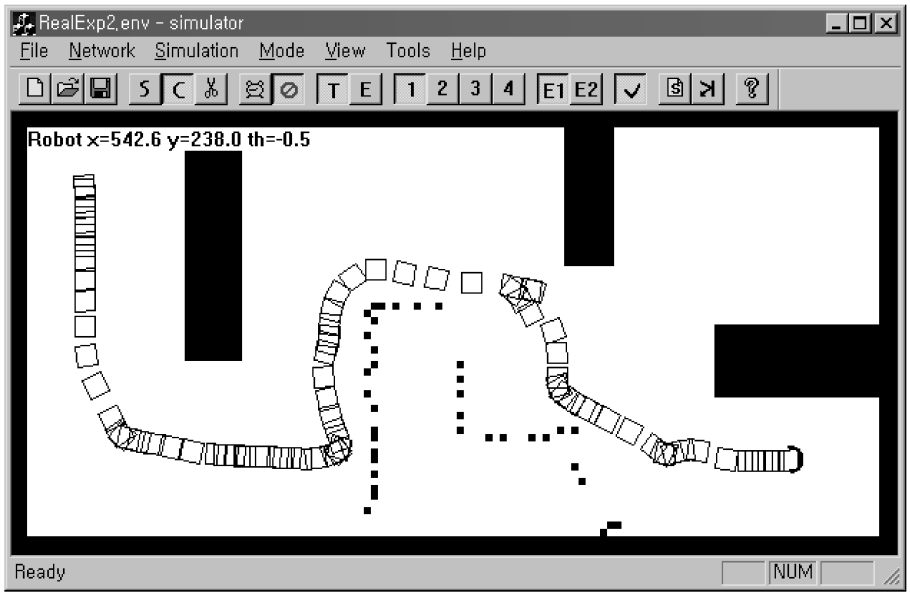


그림 4.44: 센서 정보에 의한 가상 환경의 변화 (Local site)

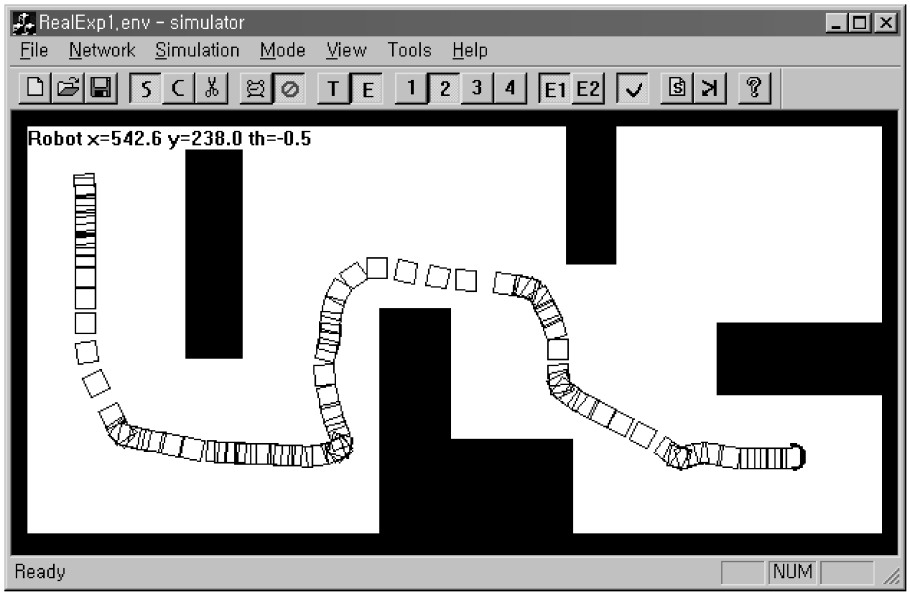


그림 4.45: Remote site의 실제 로봇의 이동 경로

5. 실험

본 장에서는 4장에서 제안한 인터넷 제어 구조에 대해 실험을 수행한다. 완성된 인터넷 기반 퍼스널 로봇 시스템이 없기 때문에, 같은 기능을 수행할 수 있는 시스템으로 실험 환경을 구성한다. 본 장은 실험 환경에 대한 설명과 실험 결과에 대해 언급한다.

5.1 실험 환경

2장에서 설명한 인터넷 기반 퍼스널 로봇 시스템을 구현하기 위해서 그림 5.1과 같이 실험 환경을 구성한다. 로봇의 절대 좌표는 높은 곳에 위치한 CCD 카메라를 통해 얻고, 비젼 시스템에서 구한 로봇의 절대 좌표는 직렬 통신을 이용하여 로봇을 직접 무선으로 제어하는 컴퓨터로 전송된다. Remote site와 local site는 하나의 컴퓨터 내에서 구현되며, local site의 시뮬레이터는 반사기(Reflector)를 통해 remote site의 시뮬레이터로 접속된다. Remote site의 시뮬레이터는 비젼 시스템을 통해 얻은 로봇의 절대 좌표와 local site로부터 전송되는 로봇의 제어 입력을 이용하여 로봇을 제어한다. 로봇의 제어 입력은 RF 통신을 이용하여 전송한다.

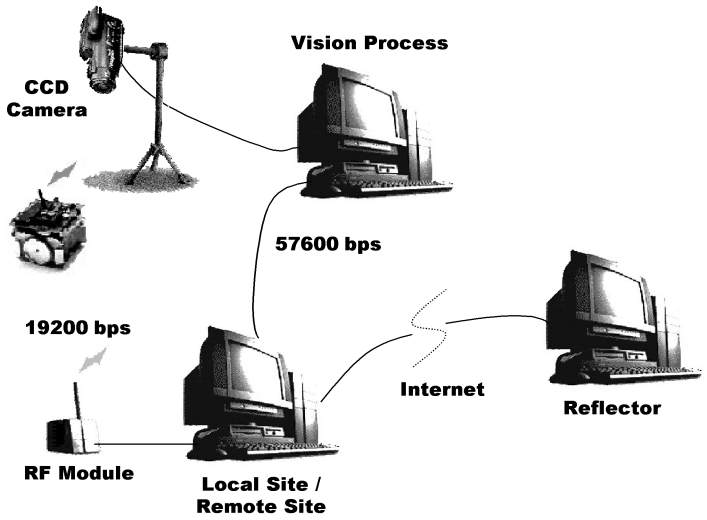


그림 5.1: 실험을 위한 전체 시스템의 구조

표 5.1은 실험에서 사용한 로봇과 비전 시스템의 사양을 나타내고, 표 5.2는 실험시 설정한 매개변수를 나타낸다.

Robot	Size	$7.5 \times 7.5 \times 7.5\ cm$
	Weight	$430\ g$
	CPU	ATMEL 89C52 11.0592 MHz 32KB ROM, 32KB RAM 2 Motion Controllers (LM629)
	Motors	DC motor 130:1 Gear ratio 125 pulse/rev resolution
	Power	9.6 V
	RF Communication	418 MHz / 433 MHz 19200 bps
Vision System	Vision Process	$\pm 1cm$ resolution 30 frames/sec
	Serial Communication	57600 bps

표 5.1: 로봇 및 비젼 시스템 사양

Simulator	Sampling time	$T = 0.4$ sec
Robot	Sampling time	$T_R = 50\ msec$
	Maximum velocity	$v_{\max} = 20\ cm/s$
	Maximum angular velocity	$\omega_{\max} = 2\ rad/s$
	Maximum acceleration	$a_{\max} = 4\ cm/s^2$
	Maximum angular acceleration	$\alpha_{\max} = 0.3\ rad/s^2$
	Size of Body	$W = 7.5\ cm$
	Length between two wheels	$L = 7\ cm$
	Radius of wheel	$R = 2\ cm$
Environment	Size	$130 \times 90\ cm$

표 5.2: 실험을 위한 매개변수 값 설정

5.2 실험 결과 및 분석

그림 5.2부터 그림 5.15까지는 인터넷 제어 구조에 따른 실험 결과를 나타낸다. 실험은 4장의 모의 실험에서와 마찬가지로 환경의 왼쪽 상위 지점에서 로봇이 출발하여 오른쪽 하위 지점으로 도착하는 것을 목표로 한다. 실험 결과는 대체로 시뮬레이터의 가상 로봇 모델과 실제 로봇 모델의 불일치에 의해 약간의 오차를 보인다.

인터넷 제어 구조 I의 경우 local site의 실험 결과에서 불연속 점이 상당히 많이 발생한다. 빈번한 불연속 점의 발생은 사용자가 시뮬레이터의 가상 로봇을 조종하는데 큰 어려움을 준다. 실제 로봇의 이동 궤적을 보면 로봇의 움직임에 있어서 불필요한 주행이 발생하고 있음을 확인할 수 있다. 사용자가 원하지 않는 움직임이 많이 나타난다는 것은 사용자가 로봇을 제어하는데 있어서 많은 어려움을 느낀다는 것을 의미한다. 인터넷 시간 지연이 샘플링 주기보다 작을 경우에는 비교적 가상 로봇과 실제 로봇 간의 경로 오차가 작다.

인터넷 제어 구조 II의 경우 인터넷 제어 구조 I에 비해 로봇의 불필요한 주행이 감소했음을 알 수 있다. 이론적으로 인터넷 제어 구조 II는 가상 로봇과 실제 로봇 간의 경로 오차가 나타나지 않는다. 실험 결과에서는 가상 로봇과 실제 로봇의 모델 차이, 또는 인터넷 시간 지연이 샘플링 주기보다 클 경우 정지 제어 입력에 대한 감가속도의 작용 등을 들 수 있다. 그림 5.9의 결과에서 보면 인터넷 시간 지연에 의한 가상 로봇과 실제 로봇간의 움직임에 대한 시간차는 줄어들지 않는다.

인터넷 제어 구조 III은 사용자가 로봇을 제어함에 있어서 용이하다는 것을 실험 결과를 통해 확인할 수 있다. 인터넷 제어 구조 II에서 발생한 가상 로봇과 실제 로봇의 움직임에 대한 시간차도 거의 나타나지 않는다. 가상 로봇과 실제 로봇의 경로 오차는 두 로봇의 모델간의 차이와 로봇 제어에 사용된 큰 샘플링 주기에 의한 것이다.

시뮬레이터의 가상 로봇 모델은 동역학을 고려하지 않았기 때문에 실제 로봇과는 큰 차이를 보인다. 가상 로봇에서도 실제 로봇의 동역학을 고려한다면 실험 결과에서 발생한 오차를 상당히 줄일 수 있다.

인터넷 제어 구조 I에 대한 실험 결과

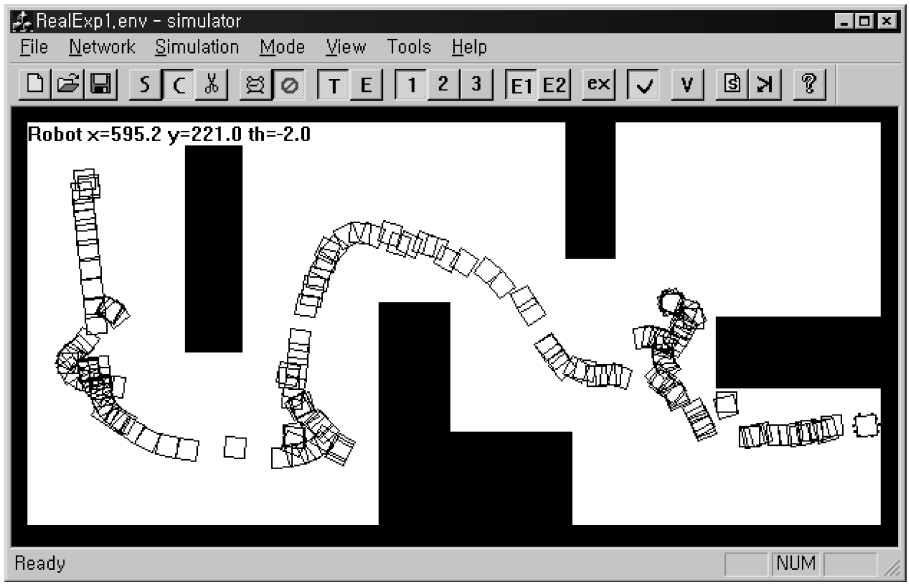


그림 5.2: Local site의 가상 로봇 이동 경로 (인터넷 제어 구조 I)

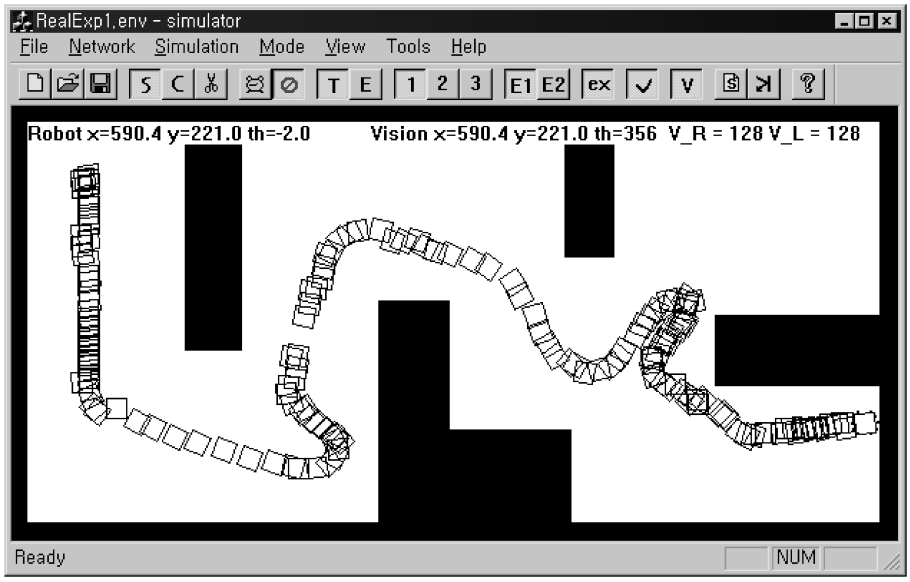


그림 5.3: Remote site의 실제 로봇 이동 경로 (인터넷 제어 구조 I)

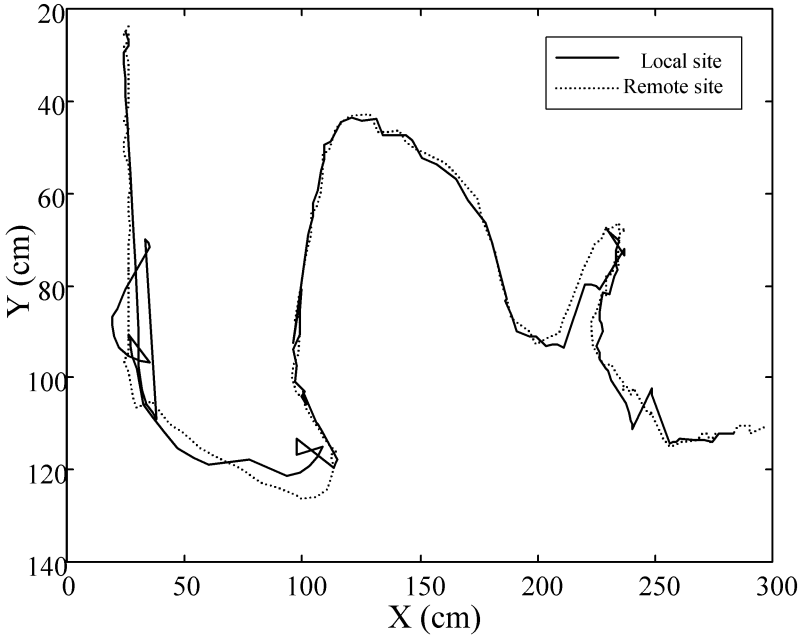


그림 5.4: 가상 로봇과 실제 로봇의 경로 오차 (인터넷 제어 구조 I)

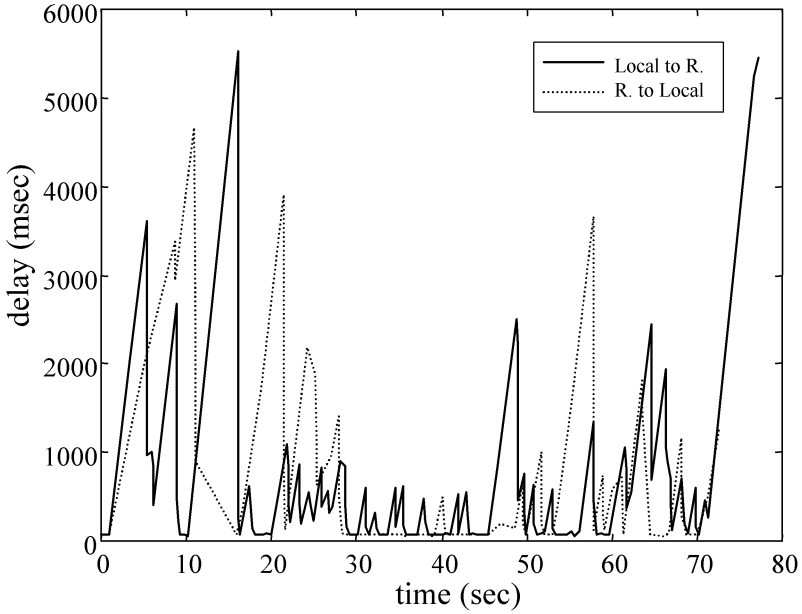


그림 5.5: 인터넷 제어 구조 I 모의 실험 중 인터넷 시간 지연

인터넷 제어 구조 II에 대한 실험 결과

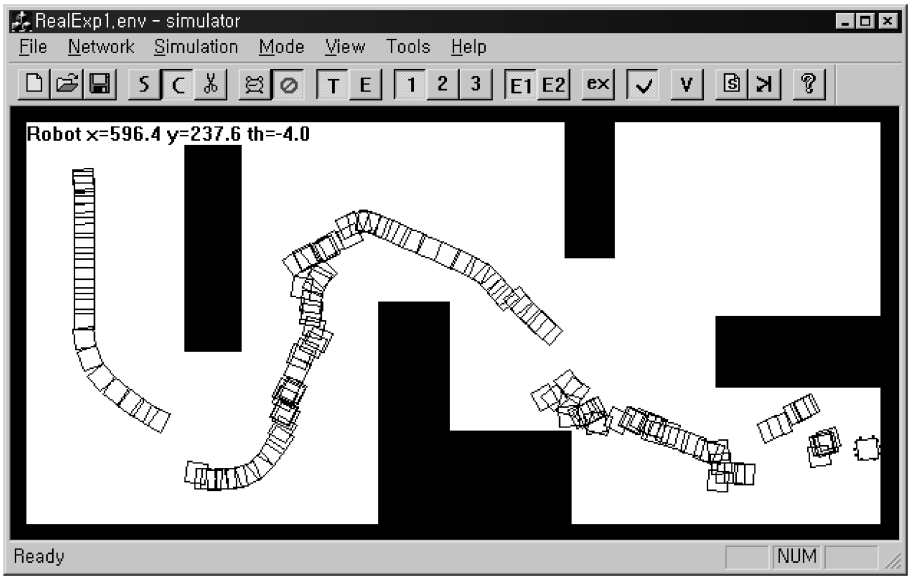


그림 5.6: Local site의 가상 로봇 이동 경로 (인터넷 제어 구조 II)

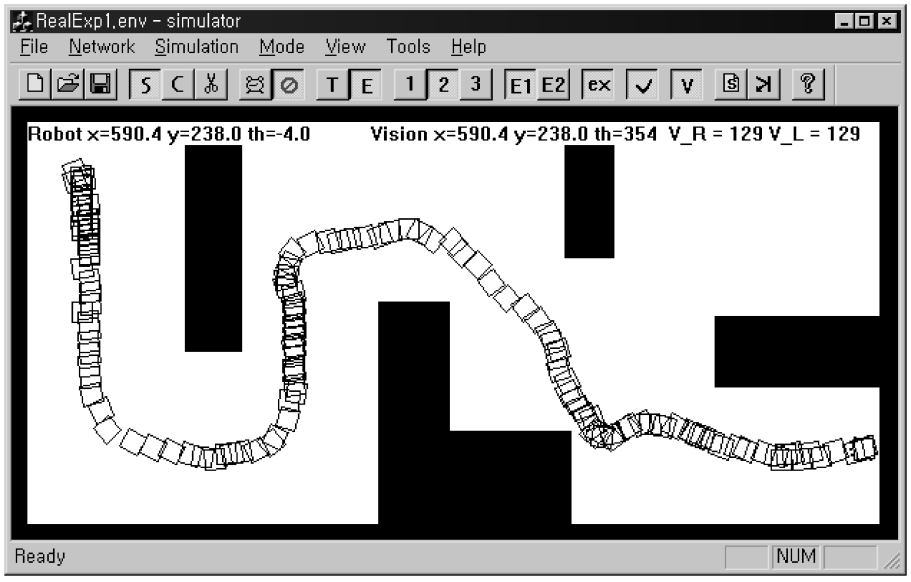


그림 5.7: Remote site의 실제 로봇 이동 경로 (인터넷 제어 구조 II)

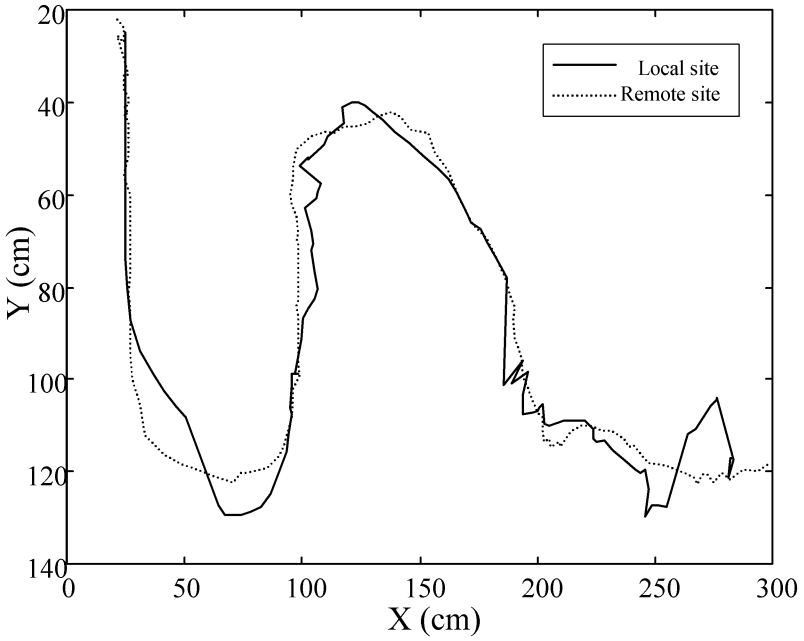


그림 5.8: 가상 로봇과 실제 로봇의 경로 오차 (인터넷 제어 구조 II)

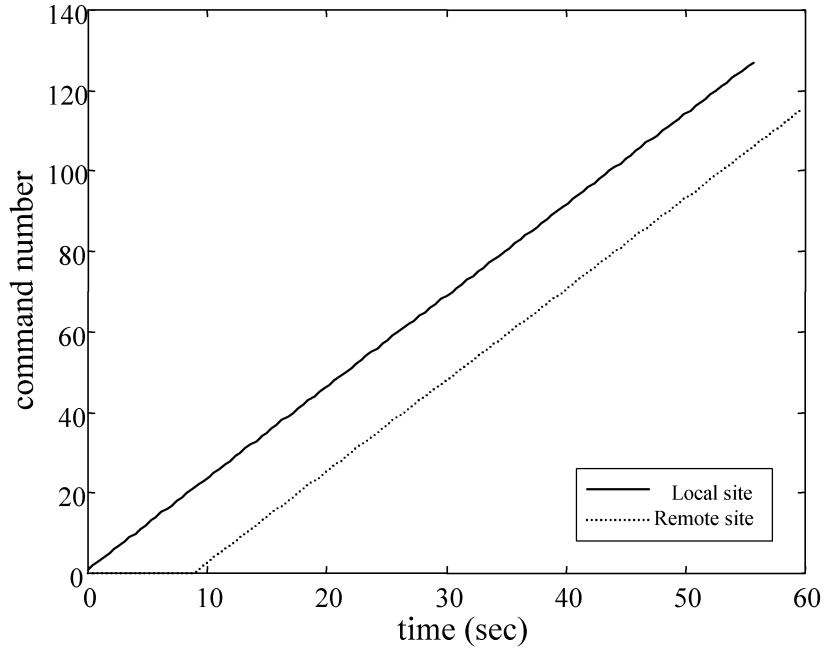


그림 5.9: 가상 로봇과 실제 로봇의 시간차 (인터넷 제어 구조 II)

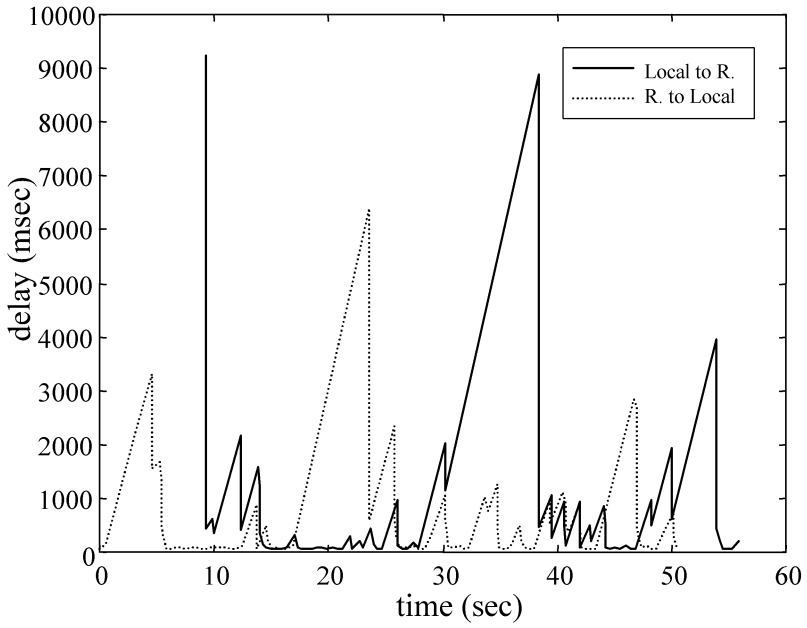


그림 5.10: 인터넷 제어 구조 II 모의 실험 중 인터넷 시간 지연

인터넷 제어 구조 III에 대한 실험 결과

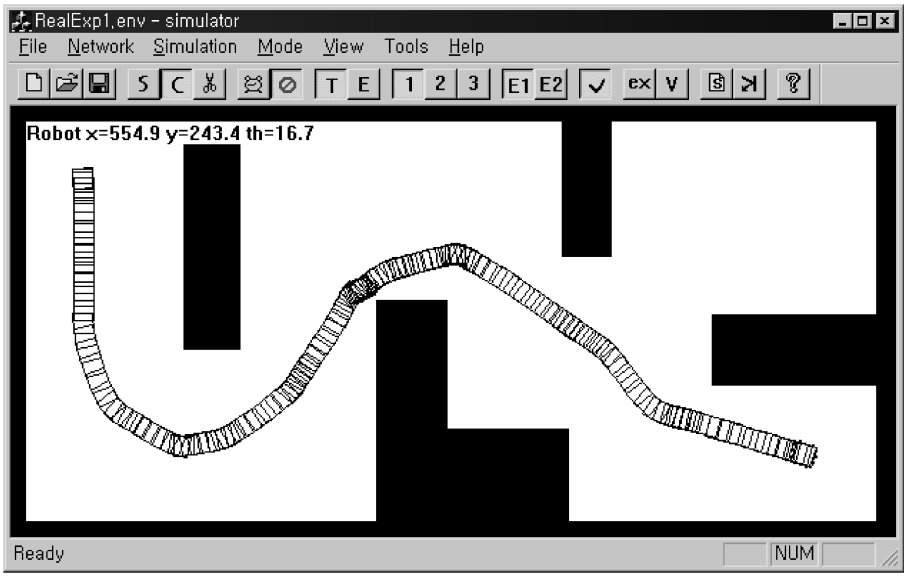


그림 5.11: Local site의 가상 로봇 이동 경로 (인터넷 제어 구조 III)

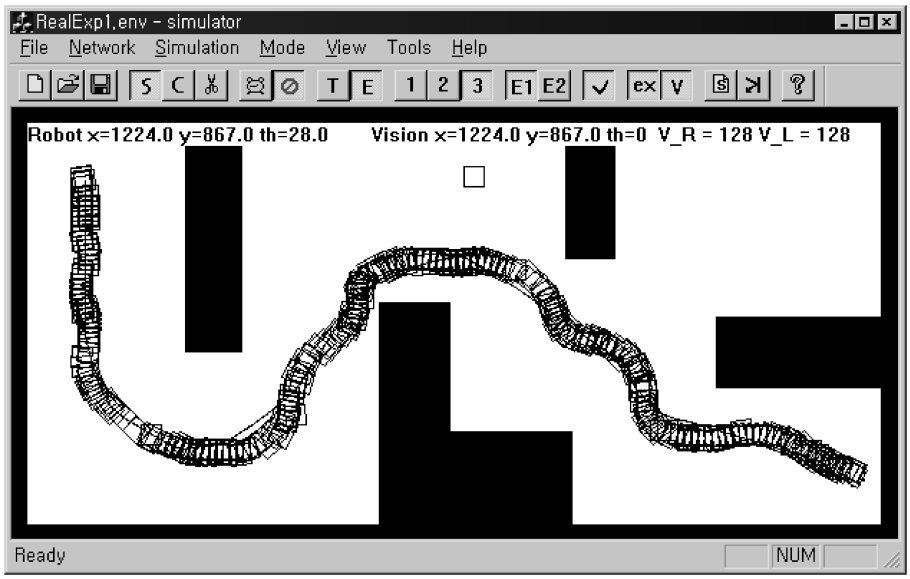


그림 5.12: Remote site의 실제 로봇 이동 경로 (인터넷 제어 구조 III)

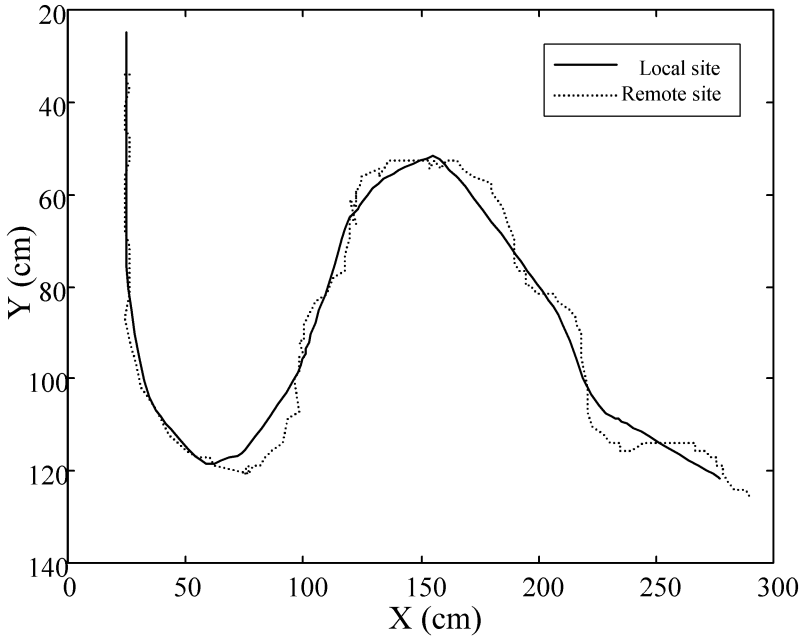


그림 5.13: 가상 로봇과 실제 로봇의 경로 오차 (인터넷 제어 구조 III)

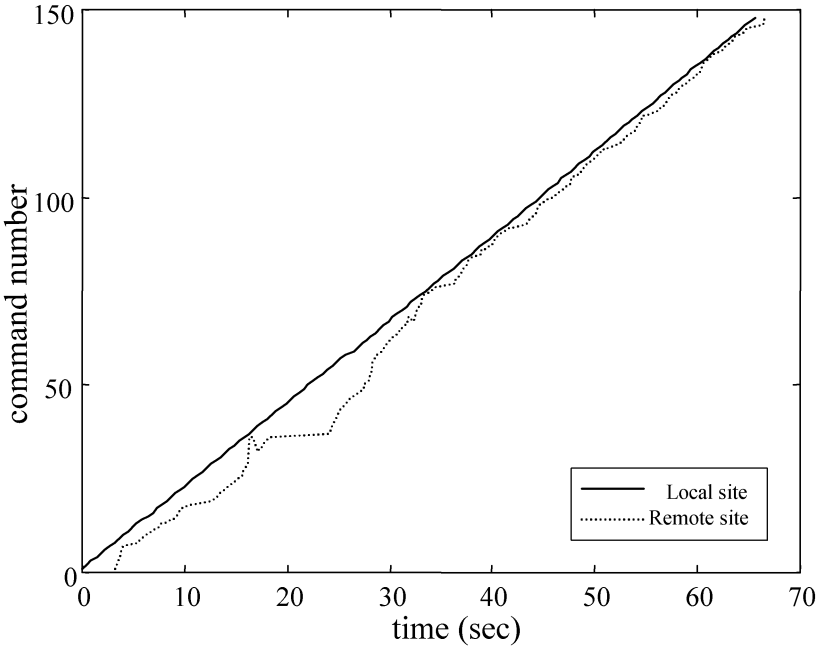


그림 5.14: 가상 로봇과 실제 로봇의 시간차 (인터넷 제어 구조 III)

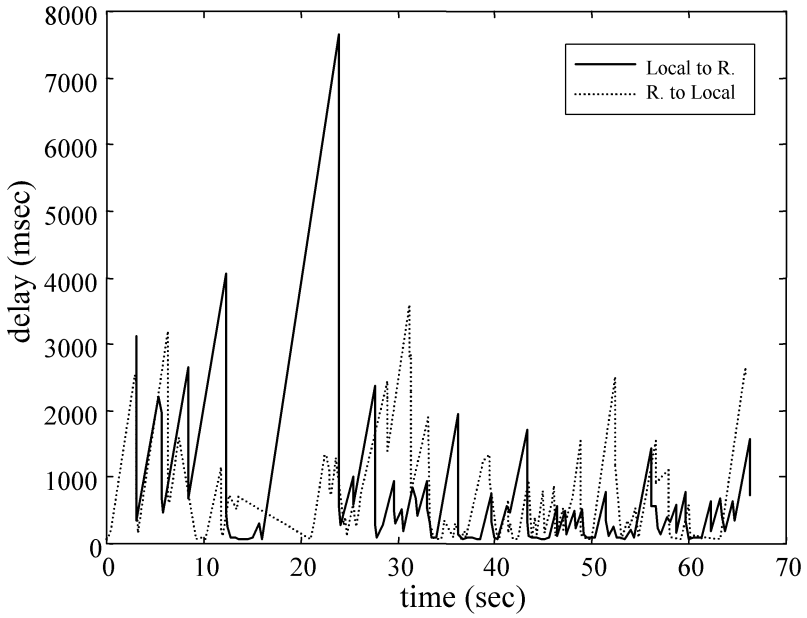


그림 5.15: 인터넷 제어 구조 Ⅲ 모의 실험 중 인터넷 시간 지연

6. 결론 및 추후 과제

본 논문에서는 인터넷 기반 퍼스널 로봇 시스템에 적합한 제어 구조를 제안하였고, 모의 실험 및 실제 로봇을 이용한 실험을 수행하였다. 인터넷 기반 퍼스널 로봇 시스템에 대한 새로운 개념을 도입하고, 인터넷 기반 제어를 위한 기본 구조를 제안하였다. 또한 인터넷 제어를 구현하기 위해 필요한 인터페이스의 두 가지 구조를 제안하였다.

인터넷 시간 지연의 분포를 조사하여 그 특성을 분석했고, 인터넷 시간 지연이 제어 입력에 미치는 영향에 대해서 살펴보았다. 인터넷을 통한 제어 입력이 샘플링 주기보다 큰 시간 지연을 갖게 될 경우, 제어 입력 정보가 손실된다는 것을 인터넷 제어 구조 I의 모의 실험을 통해 확인할 수 있었다. 인터넷 제어 구조 II에서 명령 처리 필터(Command Filter)의 개념을 새로 도입함으로써 시간 지연에 의한 제어 입력 정보의 손실을 막을 수 있었다. 인터넷 제어 구조 II에서 가상 로봇과 실제 로봇의 움직임에 있어서 인터넷 시간 지연에 의한 시간차가 줄어들지 않는 문제는 인터넷 제어 구조 III에서 경로 생성기와 경로 추종 제어기를 추가함으로써 해결이 가능하였다. 실제 환경의 변화에 의해 가상 환경과 차이가 발생하게되는 문제는 되먹임 받은 로봇의 센서 정보를 이용하여 시스템 구조를 제안함으로써 해결할 수 있었다. 센서 정보를 이용하여 장애물의 여부를 판별하는 방법, 판별된 장애물이 이미 가상 환경에 있는지의 여부를 확인하는 방법, 그리고 새로운 장애물일 경우 가상 환경에 장애물 정보를 추가시키는 방법 등을 제안하였다. 환경 변화에 대해서 제안된 구조는 모의 실험을 통해 그 성능을 확인하였다. 마지막으로 이동 로봇을 이용한 실험을 통해서 제안된 인터넷 직접 제어(direct control) 구조의 성능을 확인할 수 있었다.

본 논문에서 구상한 비전 시스템을 장착한 인터넷 기반 퍼스널 로봇을 실제로 구현하고, 제안된 인터넷 제어 구조를 로봇의 동력학까지 고려하여 적용시키는 연구가 계속되어야 할 것이다. 독립된 응용 프로그램으로 구현된 인터넷 인터페이스는 사용자가 어느 곳에서나 쉽게 사용할 수 있도록 자바, CGI를 이용한 웹 기반 인터페이스로 다시 구현하고, 사용자의 편의를 위한 감독 제어(supervisory control) 기능을 추가해야 할 것이다.

참고 문헌

[1] R. Volpe, J. Balaram, T. Ohm and R. Ivlev, "The Rocky 7 Mars Rover Prototype," in *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pp. 1558-1564, Nov. 1996.

[2] R. Oboe and P. Fiorini, "A Design and Control Environment for Internet-Based Telerobotics," *Int. Journal of Robotics Research*, vol. 17, no. 4, pp. 433-449, Apr. 1998.

[3] K. Brady and T. J. Tarn, "Internet-Based Remote Teleoperation," in *Proc. IEEE Int. Conf. Robot. Automat.*, pp. 65-70, May 1998.

[4] C. Sutter and J. Wiegley, "Desktop Teleoperation via the World Wide Web," in *Proc. IEEE Int. Conf. Robot. Automat.*, pp. 654-659, May 1995.

[5] P. G. Backes and G. K. Tharp, "The Web Interface for Telescience (WITS)," in *Proc. IEEE Int. Conf. Robot. Automat.*, pp. 411-417, Apr. 1997.

[6] K. Taylor and B. Dalton, "Issues in Internet Telerobotics," in *Int. Conf. on Field and Service Robotics*, Dec. 1997.

[7] T. M. Chen and R. C. Luo, "Remote Supervisory Control of An Autonomous Mobile Robot Via World Wide Web," in *Proc. IEEE Int. Symposium on Industrial Electronics*, vol. 1, pp. ss60-ss64, July 1997.

[8] T. M. Chen and R. C. Luo, "Multisensor Based Autonomous Mobile Robot Through Internet Control," in *Proc. of Int. Conf. on Industrial Electronics, Control, and Instrumentation*, pp. 1248-1253, Aug. 1997.

[9] M. Mitsuishi, T. Hori and T. Nagao, "Predictive, Augmented and Transformed Information Display for Time Delay Compensation in Tele-Handling/Machining," in *Proc. IEEE Int. Conf. Robot. Automat.*, pp.

45-52, May 1995.

[10] Y. Yokokohji, A. Ogawa, H. Hasunuma and T. Yoshikawa, "Operation Modes for Cooperating with Autonomous Functions in Intelligent Teleoperation Systems," in *Proc. IEEE Int. Conf. Robot. Automat.*, vol. 3, pp. 510-515, May 1993.

[11] R. J. Anderson and M. W. Spong, "Bilateral Control of Teleoperators with Time Delay," *IEEE Trans. Automat. Contr.*, vol. 34, no. 5, pp. 494-501, May 1989.

[12] 이형기, "모델링 불확실성과 시간지연이 있는 원격조작기의 강인한 양방향 제어," 박사 학위 논문, KAIST, 1998.

[13] 이하섭, "웹상에서의 Telerobotics 기술을 이용한 원격 전시 시스템," 석사 학위 논문, KAIST, 1997.

[14] M. R. Stein and R. P. Paul, "Operator Interaction, for Time-Delayed Teleoperation, with a Behavior-Based Controler," in *Proc. IEEE Int. Conf. Robot. Automat.*, pp. 231-236, May 1994.

[15] Y. J. Cho, K. Tanie, P. Akella and T. Kotoku, "Command Sequence Replanning Using Discrete-Event-Based Task Model in Tele-Robotic Part Mating," in *Proc. IEEE Int. Conf. Robot. Automat.*, pp. 647-653, May 1995.

[16] T. J. Tarn, A. K. Bejczy and N. Xi, "Motion Planning in Phase Space for Intelligent Robot Arm Control," in *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pp. 1507-1514, 1992.

[17] E. Paulos and J. Canny, "Designing Personal Tele-embodyment," in *Proc. IEEE Int. Conf. Robot. Automat.*, pp. 3173-3178, May. 1998.

[18] J. OTA and T. ARAI, "A Hybrid Technique to Supply Indoor Service Robots," in *Proc. IEEE Int. Conf. Robot. Automat.*, pp. 89-94, May. 1998.

[19] P. Kopacek, "Service Robots Present Situation and Future Trends," in *Proc. of 5th Int. Workshop on Robotics in Alpe-Adria-Danube Region*, pp. 29-33, June 1996.

[20] H. Hirukawa, T. Matsui and H. Onda, "Prototypes of Teleoperation Systems via a Standard Protocol with a Standard Human Interface," in *Proc. IEEE Int. Conf. Robot. Automat.*, pp. 1028-1033, Apr. 1997.

[21] 김용재, "가상 구심력장과 중력장을 이용한 이동 로봇의 통합 항법 방식," 석사 학위 논문, KAIST, 1998.

[22] Y. F. Zheng, (ed.), *Recent Trends in Mobile Robots*, World Scientific, 1993.

[23] L. E. Aguilar M., P. Souères, M. Courdesses and S. Fleury, "Robust Path-following Control with Exponential Stability for Mobile Robots," in *Proc. IEEE Int. Conf. Robot. Automat.*, pp. 3279-3284, May 1998.

[24] C. Canudas and O. J. Sørdalen, "Exponential Stabilization of Mobile Robots with Nonholonomic Constaints," *IEEE Trans. Automat. Contr.*, vol. 37, no. 11, pp. 1791-1797, Nov. 1992.

[25] S. M. Lavalle and R. Sharma, "On Motion Planning in Changing, Partially Predictable Environments," *Int. J. Robotics Research*, vol. 16, no. 6, pp. 775-805, Dec. 1997.

[26] J. Borenstein and Y. Koren, "The Vector Field Histogram - Fast Obstacle Avoidance for Mobile Robots," *IEEE J. Robot. Automat.*, vol. 7, no. 3, pp. 278-288, June 1991.

[27] A. Fujimori, P. N. Nikiforuk and M. M. Gupta, "Adaptive Navigation of Mobile Robots with Obstacle Avoidance," *IEEE Trans. Robot. Automat.*, vol. 13, no. 4, pp. 596-602, Aug. 1997.

감사의 글

이 논문이 완성되기까지 물심양면으로 도움을 주신 많은 분들에게 진심으로 감사를 드립니다. 먼저 2년 동안 언제나 곁에서 지켜봐 주시고 값진 조언과 격려를 아끼지 않으셨던 지도 교수님이신 김종환 교수님께 깊은 감사를 드립니다. 바쁜 일정에도 불구하고 많은 관심으로 논문 심사를 해주신 정명진 교수님과 임종태 교수님께도 깊은 감사를 드립니다.

큰 형 같은 동균이형, 항상 든든함을 느끼게 해주셨던 정열이형, 일에 몰두하는 법을 몸소 보여주신 현식이형, 광춘이형, 성실함의 표본 현이형, 진정한 유머를 보여주시는 정민이형, 옆에서 함께 고민하고 조언을 아끼지 않았던 신이형, 항상 명령한 치호형, 모든 일에 최선을 다하는 명진이형, 홍수형, 여러 밤을 함께한 동한이형, 용재형, 격려 편지를 보내주신 인환이형, 힘든 일도 마다하지 않는 문수형, 승은이, 그리고 논문을 쓰면서 서로에게 많은 힘이 되었던 귀홍이, 상호, 광희에게 고마운 마음을 전합니다. 언제나 많은 도움을 주신 Prahlad 박사님에게도 감사를 드립니다.

바쁘다는 핑계로 자주 만나지도 못했지만 항상 이해해주고 격려해주었던 준우와 유석이에게도 고맙다는 말을 전합니다. 가까운 곳에서 힘이 되어준 지철이와 상길이, 룸메이트 진성이, 정이와 필순이를 비롯한 미라지 동아리 후배들, 그리고 8년 동안 한 울타리에서 동고동락한 대전 과학 고등학교 8기들에게도 고마움을 전합니다.

한 마디의 불평도 없이 언제나 따뜻한 마음으로 함께 걱정해주고, 격려해주고, 챙겨준 명진이에게 사랑과 감사의 말을 전합니다.

마지막으로 저를 위해 항상 기도해주시고 보살펴 주시는 사랑하는 부모님과 먼 곳에서도 큰 힘이 되어준 누나, 형에게 무한한 감사를 드립니다.

이 력 서

성　명 : 한 국 현

생년월일 : 1975년 7월 11일

출 생 지 : 서울특별시

본　적 : 서울특별시

학　력

1993.3 - 1997.2　한국과학기술원 전기 및 전자공학과 (B.S.)

1997.3 - 1999.2　한국과학기술원 전기 및 전자공학과 (M.S.)