유전자 알고리즘을 이용한 네트워크 기반 제어 시스템의 원격 제어기 설계

Remote Controller Design of Networked Control System Using Genetic Algorithm

이 경 창, 이 석
(Kyung Chang Lee and Suk Lee)

Abstract : As many sensors and actuators are used in automated systems, various industrial networks are adopted for digital control system. In order to take advantages of the networking, however, the network implementation should be carefully designed to satisfy real-time requirements considering network delays. This paper presents the implementation scheme of a networked control system via Profibus-DP network. More specifically, the effect of the network delay on the control performance was evaluated on a Profibus-DP testbed, and a GA-based PID tuning algorithm is proposed to design controllers suitable for networked control systems.

Keywords : networked control system, remote controller, Profibus-DP, fieldbus, network delay, PID tuning, genetic algorithm

I. 서론

최근 마이크로 프로세서의 기능과 성능이 발전함에 따라, 많은 제어 시스템이 디지털 제어 시스템으로 설계되고 있다. 특히, 산업용 컴퓨터를 이용한 디지털 제어 시스템은 뛰어난 유연성과 적응성으로 인하여, 조립 자동화나 공정 자동화 분야에서 활발하게 적용되고 있는 추세이다. 또한, 향상된 시스템 성능을 얻기 위하여 더 많은 수의 센서, 구동기, PLC (Programmable Logic Controller), 산업용 컴퓨터 등과 같은 필드기기들(field devices)이 사용됨에 따라, 제어 시스템에서 처리되어야 할 데이터량이 급속하게 증가되었다.

그러나, 두 개의 필드기기를 직접 연결하는 점대점(point-to-point) 연결 방식은 너무 많은 배선을 필요로 하기 때문에, 신뢰성 확보가 어려울 뿐만 아니라, 시스템의 유지 및 보수가 복잡하게 되었다. 이러한 문제의 해결 방법으로서, 점대점 연결 방식을 공유된 전송매체를 사용하는 산업용 네트워크(industrial network)로 변환시키려는 연구들이 활발하게 진행되고 있다. 산업용 네트워크에 대한 연구의 결과, 제어 시스템은 성능 향상과 단순화, 신뢰성 증대 등과 같은 장점을 얻을 수 있게 되었다[1].

일반적으로, 산업용 네트워크에서 전송되는 데이터는 산발적 실시간과 주기적 실시간, 그리고 비실시간 데이터로 구분되며, 이러한 데이터들은 단일한 전송 매체를 통하여 동시에 전송될 수 있어야 한다. 만약, 이러한 데이터들이 제대로 관리되지 못하면, 즉 산발적 실시간 데이터나 주기적 실시간 데이터가 정해진 한계치 내에 전송되어야 한다는 실시간 요구 조건이 만족되지 못하면, 시스템의 성능이 저하될 뿐만 아니라, 시스템에 치명적인 오류가 발생하게 된다. 따라서, 디지털 제어 시스템을 위한 산업용 네트워크에서는, 필드기기에서 요구되는 실시간 요구 조건이 만족될 수 있어야 한다[2]-[4].

접수일자 : 2001. 5. 11., 수정완료 : 2001. 9. 11.
이경창, 이 석 : 부산대학교 기계공학부 및 기계기술연구소(gclee@pnu.edu/slee@pnu.edu)

이러한 목적으로, 1980년대 후반, 필드기기들의 실시간 요구조건을 만족시킬 수 있는 산업용 네트워크, 즉 필드버스(fieldbus)가 개발되었으며, Profibus, WorldFIP, Fieldbus Foundation, CAN, LonWorks 등과 같은 프로토콜이 개발되었다. 최근에는 IEC에 의해 Profibus, Fieldbus Foundation 등을 포함하는 IEC 61158이 국제 표준으로 제정되었다[5].

특히, 최근 들어 필드버스는 그림 1과 같은 분산 로봇 시스템(fully distributed robot system)[6]과 같은 네트워크 기반 제어 시스템(networked control system)에도 활발하게 응용되고 있는 추세이다[7][8]. 그림에서, 구동기나 센서, 제어기는 필드버스에 연결되어 있으며, 제어기는 로봇의 원활한 동작을 위하여 센서로부터 로봇 팔의 위치 정보를 수신하거나 구동기로 제어 정보를 송신한다. 만약, 이러한 제어 시스템에서 로봇의 위치 정보나 제어기의 제어 정보가 주어진 샘플링 시간 내에 제어기에 도달되지 못하게 되면, 제어 성능이 나빠질 뿐만 아니라, 최악의 경우에는 로봇의 궤적이 정해진 영역을 벗어나 다른 물체와 충돌할 수 있게 된다. 따라서, 제어 시스템에 필드버스를 도입하는 경우, 제어 신호와 센서 정보가 주어진 샘플링 시간 내에 전송될 수 있도록 세심한 주의가 요구된다.

본 논문에서는 필드기기의 실시간 요구 조건을 만족시킬 수 있는 네트워크 기반 제어 시스템의 구조와 유전자 알고

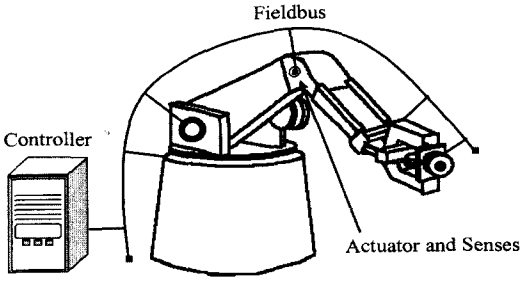


그림 1. 네트워크 기반 제어 시스템의 예.
Fig. 1. Example of networked control system.

리즘(Genetic Algorithm, GA)을 이용한 원격 제어기의 설계 방법을 제안하고, 이를 이용한 네트워크 기반 제어 시스템의 성능을 실험적으로 평가하였다. 본 논문은 서론을 포함하여 6장으로 구성되어 있다. II장에서는 IEC 국제표준의 하나로서, 제어 시스템에 적합한 폴링(polling) 방식을 사용하는 Profibus-DP에 대하여 설명하였다. III장에서는 Profibus-DP를 이용한 네트워크 기반 제어 시스템의 설계 방법을 제시하고, 네트워크 지연(network delay)의 원인에 대하여 규명하였다. IV에서는 GA를 이용한 원격 제어기 설계 방법에 대하여 제안하였고, V장은 제안된 설계 방법으로 구성된 DC 모터 원격 제어 시스템의 실험적 평가 결과를 담고 있다. 마지막으로, VI장에서는 결론을 제시하였다.

II. Profibus-DP 프로토콜

1. Profibus-DP 프로토콜의 구조

Profibus-DP는 공정 시스템이나 제어 시스템에서의 적용을 위하여 개발되었으며, 이러한 응용들에서 필요한 실시간 요구 조건을 만족시키기 위하여, 그림 2와 같이 OSI 참조 모델 7개 계층 중 1, 2계층인 PHY(PHYsical layer)와 FDL(Fieldbus Data Link layer)만을 사용한다. 또한, FDL 계층에서는 전송지연이 크게 변화하는 것을 피하기 위하여 매체 접속 제어 (Medium Access Control) 방식으로 폴링(polling)을 사용한다. 이에 추가하여, Profibus-DP는 네트워크 관리 기능을 위하여 FMA1/2(Fieldbus MAnagement layer 1 & 2)와 사용자 인터페이스(user interface)와 FDL 간의 데이터 교환을 위하여 DDLM(Direct Data Link Mapper)을 사용한다[9].

Profibus-DP의 전송 매체로는 RS485가 사용되며, 전송속도는 9.6Kbps에서 12Mbps까지 지원된다. Profibus-DP에 접속되는 스테이션들은 마스터 스테이션(master station)과 슬레이브 스테이션(slave station)으로 구분된다. 마스터는 미리 정해져 있는 폴링 주기 동안 슬레이브에게 폴링을 수행하는 스테이션으로, PLC나 산업용 컴퓨터와 같은 중앙 제어기의 역할을 한다. 슬레이브는 주변 기기로부터 입력 정보를 수집하거나, 주변 기기에 출력 정보를 보내는 필드기기로서의 역할을 수행한다. 일반적으로, 마스터는 슬레이브에게 데이터를 전송하거나, 전송을 요청할 수 있으나, 슬레이브는 마스터의 요청에 의해서만 통신에 참여할 수 있다.

그림 3은 Profibus-DP의 일반적인 통신 모델을 나타내고 있다. 통신 프로세스를 가진 사용자 인터페이스는 FDL과의 원활한 통신을 위하여 DPRAM(Dual Port RAM)을 이용한다. 사용자 인터페이스에서 동작하는 모듈들은 폴링 주기마다 DPRAM의 입력 버퍼(input buffer)에 수신되어 있는 데이터를 읽거나, 전송할 데이터를 해당 모듈의 출력 버퍼(output buffer)에 기록한다. 또한, FDL은 DPRAM의 출력 버퍼로부터 데이터를 읽어 각 슬레이브로 송신하거나, 각 슬레이브로부터 응답 프레임을 수신하여 해당 DPRAM의 입력 버퍼에 기록한다. 이러한 통신 모델의 특성, 즉 응용 프로세스와 통신 기능이 DPRAM에 의하여 분리되어 있는 특성으로 인하여, Profibus-DP는 엄격한 실시간이 요구되는 시스템에 적합하다고 알려져 있다.

2. Profibus-DP의 성능

본 절에서는 Profibus-DP가 분산된 필드기기에서 요구되는 실시간 요구 조건을 만족시키는 가를 평가하기 위하여 실험 모델을 구성하고, 성능 평가를 수행하였다. 실험 모델은 마스터 역할을 하는 한 대의 컴퓨터와 슬레이브 역할을 하는 다수의 컴퓨터로 구성되어 있으며, 각 컴퓨터에는 Softing사의 PROFI-Board를 장착하였다[10].

그림 4는 Profibus-DP의 슬레이브 수에 따른 반응시간(overall response time)을 나타내고 있다. 여기에서, 반응시간이란 마스터가 첫 번째 슬레이브에게 통신 요청을 보내고 나서부터 마지막 슬레이브로부터 응답을 받을 때까지 걸린 시간으

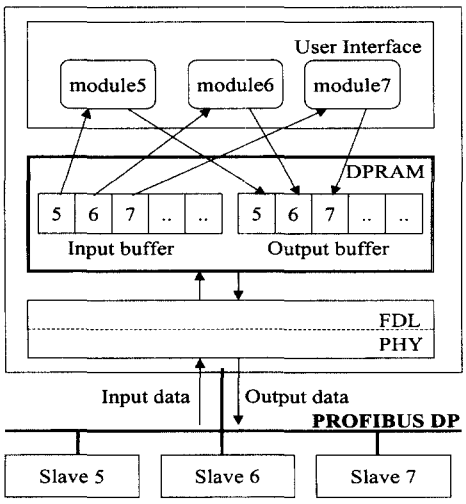


그림 3. Profibus-DP의 통신 모델.
Fig. 3. Communication model of Profibus-DP.

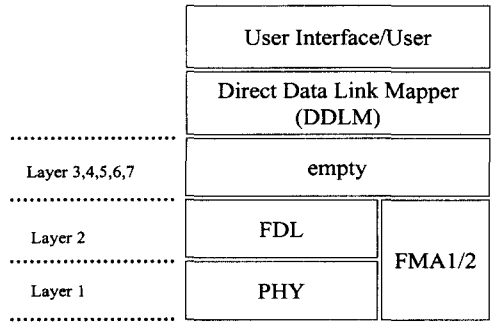


그림 2. Profibus-DP의 프로토콜 구조.
Fig. 2. Protocol architecture of Profibus-DP.

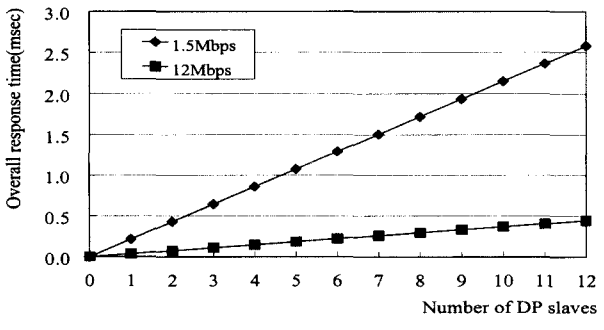


그림 4. Profibus-DP의 반응시간.
Fig. 4. Overall response time of Profibus-DP.

로 정의하였다. 이 실험에서 마스터는 슬레이브와의 폴링을 위하여 2바이트(byte)의 메시지를 보내며, 슬레이브는 2바이트의 메시지로 응답한다. 여기에서, 메시지 길이가 2바이트인 이유는 일반적으로 제어기의 제어신호와 슬레이브의 응답신호를 표현하기에 충분하기 때문이다. 또한, 각각의 슬레이브가 마스터로부터 폴링 신호를 수신하는 주기, 즉 폴링 주기는 대다수의 실시간 시스템에서 사용되는 10msec로 선정하였다[11].

그림에서 Profibus-DP의 전송속도가 1.5Mbps일 때, 1개의 슬레이브만이 통신을 하는 경우 반응시간은 215 μsec로 나타났다. 슬레이브의 수가 증가하는 경우 반응시간은 선형적으로 증가하였으며, 12개의 슬레이브가 통신을 하는 경우에도 3msec 이하로써, 매우 빠른 반응시간을 나타내고 있다.

여기에서, 슬레이브의 수와 반응시간의 관계가 선형으로 나타나는 것은 메시지의 충돌과 같은 불확실성이 없기 때문에, 하나의 슬레이브가 추가되면 일정한 반응시간만이 증가되기 때문이다. 또한, 전송속도가 12Mbps일 때, 1개의 스테이션이 통신을 하는 경우 반응시간은 37 μsec로 나타났으며, 12개의 스테이션이 통신을 하는 경우에도 반응시간은 1msec 이하로 매우 낮게 나타났다. 이러한 결과로 볼 때, 제어기와 필드기기 간의 샘플링 주기가 10msec 정도인 엄격한 실시간 시스템에서, Profibus-DP는 실시간 요구 조건을 만족시킬 수 있는 네트워크임을 알 수 있었다.

III. 네트워크 기반 제어 시스템

1. 네트워크 기반 제어 시스템의 구조

그림 5는 Profibus-DP에서의 네트워크 기반 제어 시스템의 구조를 나타내고 있다. 그림에서 마스터/슬레이브 스테이션은 입/출력 데이터 프로세스(input/output data process), 통신 프로세스 (communication process) 및 I/O 계측 프로세스(I/O instrumentation process)로 구성되어 있다. 또한, 네트워크 기반 제어 시스템에서 통신은 통신 사이클(communication cycle)과 계측 사이클(instrumentation cycle)에 의하여 이루어진다. 통신 사이클에서는 DPRAM과 통신 프로세스를 이용하여 마스터의 제어 정보와 슬레이브의 출력 정보가 서로 교환되며, 계측 사이클에서는 I/O 계측 프로세스를 통하여 플랜트 제어에 필요한 정보가 교환된다.

마스터의 동작 순서는 다음과 같다. 통신이 시작되면, 슬레이브와의 정보 교환을 위하여 DP 마스터 함수 기능을 사용하는 입/출력 데이터 프로세스가 동작된다. 입력 데이터 프로세스는 DPRAM 입력 버퍼로부터 피드백 정보를 읽어 출력 데이터 프로세스로 전달한다. 출력 데이터 프로세스 내에 구현되어 있는 플랜트의 제어기 모듈은 피드백 정보를 이용하여 플랜트의 제어 정보를 생성하며, 이를 슬레이브로 전송하기 위하여 DPRAM 출력 버퍼에 저장한다.

슬레이브의 경우에도 통신이 시작되면, 마스터와의 정보 교환을 위하여 DP 슬레이브 함수 기능을 사용하는 입/출력 데이터 프로세스가 동작된다. 입력 데이터 프로세스는 DPRAM 입력 버퍼로부터 제어 정보를 읽어 입력 데이터 프로세스로 전달한다. 출력 데이터 프로세스 내에 있는 플랜트 구동 모듈은 제어 정보를 이용하여 (계측 사이클을 따라) 플랜트를 동작시킨다. 또한, 플랜트로부터 읽은 피드백 정보를 마스터로 전송하기 위하여 DPRAM 출력 버퍼에 저장한다.

2. Profibus-DP에서의 네트워크 지연 특성

네트워크 기반 제어 시스템에서는 필드버스의 사용으로 인하여, 제어 신호 및 피드백 신호의 전송에 지연이 발생한다. 이러한 네트워크 지연은 제어기가 정수 배의 샘플링 시간 이후에 플랜트의 피드백 신호를 사용하게 함으로써, 제어 성능을 악화시키게 된다[7][8].

그림 6은 Profibus-DP에서 작동하는 네트워크 기반 제어 시스템의 각 프로세스들의 송신 및 수신 시간을 보여주고

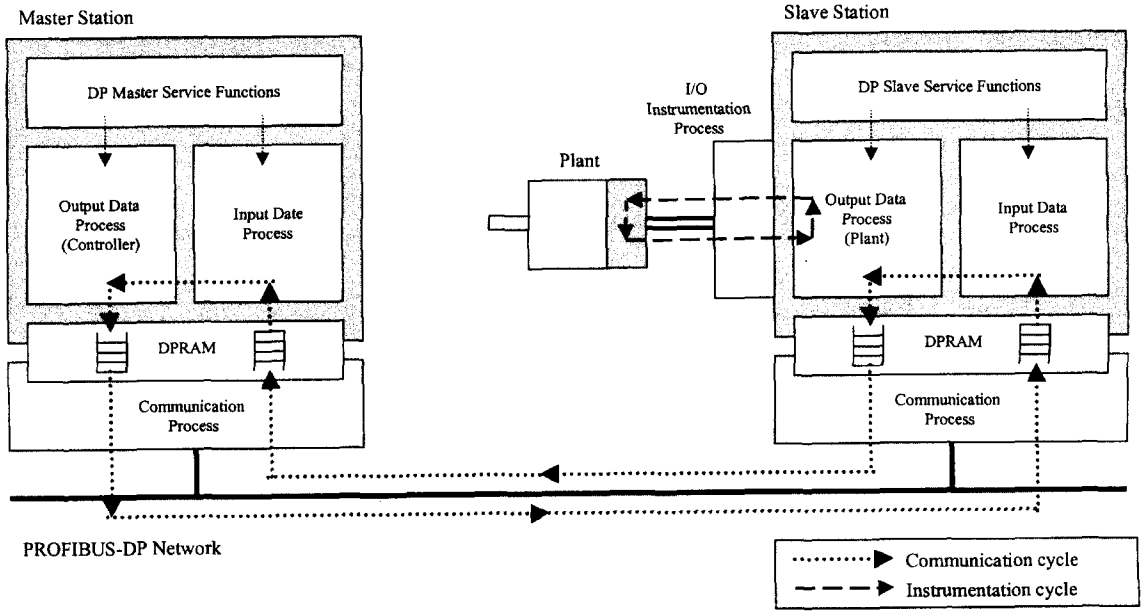


그림 5. Profibus-DP에서의 네트워크 기반 제어 시스템의 구조.
Fig. 5. Structure of networked control system using Profibus-DP.

있다. 여기서 T_{CYC}는 Profibus-DP의 통신 주기 또는 각 프로세스들의 동작 주기를 나타내며, T_P는 입출력 데이터 프로세스의 처리 시간, T_T는 통신 프로세스의 처리 시간과 제어 신호나 피드백 신호의 전송시간의 합, T_I는 계측 프로세스의 처리 시간, T_C는 제어 신호의 길이이다. 또한, 각 프로세스를 표시하는 수평선 위에 그려진 블록은 마스터에서 슬레이브로 전달되는 데이터를 나타내고 수평선 아래의 블록은 반대 방향으로 전송되는 데이터를 나타낸다. 이러한 전송 순서에서, 네트워크 지연은 다음과 같은 원인에 의하여 발생한다.

첫째, 네트워크 기반 제어 시스템의 각 프로세스들이 완벽한 동기를 이루었다고 하더라도, 각각의 프로세스가 수행되기 위하여서는 일정한 시간이 필요하고, 또한 슬레이브의 전송을 위해서는 마스터의 폴링을 기다려야 한다는 점 때문에 지연이 발생한다. 이러한 상황을 그림 6a에 나타내었는데 마스터에서 출력 데이터 프로세스가 T_P 동안 제어신호 C1을 생성하여 DPRAM에 저장하면, 통신 프로세스가 이를 T_T 동안 처리하여 전송한다. C1을 수신한 슬레이브의 통신 프로세스는 수신된 데이터를 DPRAM에 저장하고, 바로 DPRAM의 출력 버퍼에 저장되어 있던 데이터를 전송한다. 이때 전송되는 데이터는 수신된 데이터가 슬레이브의 응용 프로세스에서 처리된 상태가 아니라, 통신이 초기화될 때 저장되었던 데이터가 전송되게 된다. 그리고 나서, 슬레이브의 입력 데이터 프로세스는 T_P 동안 C1을 읽어 들여 계측 프로세스로 전달한다. 계측 프로세스는 수신된 제어 신호를 이용하여 시스템을 구동하고 센서 데이터를 측정하여 피드백 신호 F1을 생성한다. 이 피드백 신호는 지금까지의 과정의 역순으로 전송된다. 다만, 한 가지 차이점은 슬레이브의 전송은 마스터의 폴링에 의해서만 가능하기 때문에 통신 프로세스가 모든 준비를 마쳤다 하더라도 다음 통신 주기까지 지연된다는 점이다(①). 이러한 문제로 인하여, 그림 6.a에 나타나 있는 것처럼 준비된 피드백 신호의 전송은 다음 제어신호 C2가 수신된 후에야 마스터로 전송된다. 따라서, 한 제어 신호가 생성된 시점부터 그에 상응하는 피드백 신호가 수신될 때까지 $T_{CYC}+2T_P+2T_T+T_C$가 소요됨을 알 수 있다.

둘째, 네트워크 지연은 각 프로세스들간의 비동기화에 의하여 발생한다. 그림 6a에서 마스터의 입/출력 프로세스로부터 통신 요청된 제어 프레임 C1은 통신 프로세스에 도착하게 되면 바로 전송이 되도록 두 프로세스간의 동기화가 이루어져 있어야 한다. 그러나, 현실적으로 이러한 동기의 유지는 대단히 어려운 문제이기 때문에, 네트워크를 설계할 때, 비동기화에 의하여 발생할 수 있는 네트워크 지연을 고려하여야 한다. 그림 6b는 동기화가 이루어지지 않았을 경우 프레임들의 전송 순서를 나타내고 있다. 그림에서, 각 프로세스들이 모두 동시에 시작하는 것으로 가정을 하는 경우, 프로세스들의 비동기화에 의한 네트워크 지연은 마스터

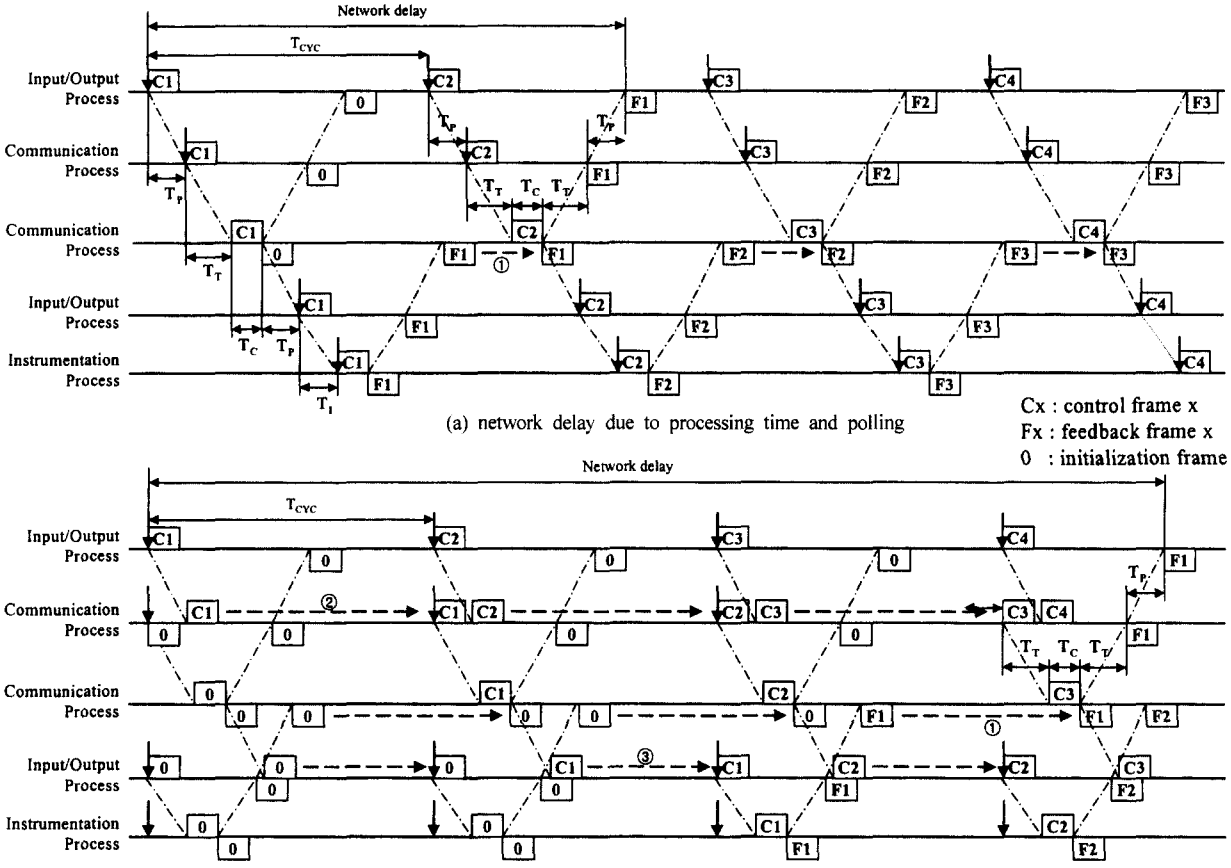


그림 6. 네트워크 지연이 발생하는 경우의 Profibus-DP의 전송 순서.
Fig. 6. Transmission sequence of Profibus-DP with network delay.

와 슬레이브 모두에서 발생한다. 제어 신호의 전송과 피드백 신호의 전송 순서는 그림 6a에서 설명한 방법과 유사하다. 다만, 마스터의 입/출력 프로세스와 통신 프로세스가 동기화가 이루어지지 않았기 때문에, 마스터의 입력 프로세스로부터 통신 요청된 C1은 즉시 전송되지 못하고, 통신 프로세스의 다음 통신 주기에 송신된다. 이로 인하여 네트워크 지연(②)이 발생한다. 또한, 슬레이브의 입/출력 프로세스와 계측 프로세스가 동기화가 이루어지지 않았기 때문에, 슬레이브의 입/출력 프로세스가 통신 프로세스로부터 제어 신호를 받게 되는 경우, 즉시 계측 프로세스로 제어 신호를 전송하지 못하고, 다음 통신 주기에 전송하게 된다. 이로 인하여, 네트워크 지연(③)이 발생한다. 따라서, 네트워크 지연은 ①, ②, ③의 원인에 의하여 한 제어 신호가 생성된 시점부터 그에 상응하는 피드백 신호가 수신될 때까지 $3T_{CYC} + T_P + 2T_T + T_C$가 소요됨을 알 수 있다.

위와 같은 이유로, Profibus-DP에서 네트워크 기반 제어 시스템을 구성하는 경우 유한한 네트워크 지연이 발생함을 확인하였다. 일반적으로, 이러한 네트워크 지연은 플랜트의 제어 성능을 악화시키기 때문에, 네트워크 지연이 최소화되도록 네트워크 기반 제어 시스템이 구성되어야 하며, 원격 제어기를 설계할 때 네트워크 지연에 대한 세심한 고려가 필요하다. 특히, 본 연구에서 설계된 Profibus-DP를 이용한 네트워크 기반 제어 시스템의 경우, 세 배 이상의 통신 주기에 해당하는 지연이 발생하는 것을 확인하였다.

IV. GA에 의한 원격 제어기 설계

유전자 알고리즘은 1962년 Holland에 의하여 소개된 적자생존의 법칙에 기반을 둔 최적화 이론으로서, 최적화 문제나 PID 튜닝과 같은 분야에서 효과적인 것으로 알려져 있다. GA는 전체 탐색 공간에서 여러 개의 스트링(string), 즉 인구(population)를 이용하여 동시에 탐색을 수행하기 때문에, 전역 최적값으로 수렴할 가능성이 클 뿐만 아니라, 미분 불가능이나 비선형, 멀티모달(multi-modal)과 같은 문제에서 강인한 특성을 발휘한다고 알려져 있다[12].

본 절에서는 네트워크 기반 제어 시스템에서 발생하는 네트워크 지연을 보상하기 위한 방법으로서, GA에 의한 원격 제어기의 설계 방법을 설명한다.

1. GA에 의한 원격 제어기의 구조

그림 7에는 GA에 의한 원격 제어기 설계 방법에 대한 개념도를 나타내었다. 그림의 아래 부분은 플랜트와 원격 제어기로 구성된 네트워크 기반 제어 시스템을 나타내고 있다. 여기에서, 플랜트와 원격 제어기는 필드버스로 연결되어 있으며, 플랜트는 Profibus-DP의 슬레이브 스테이션으로, 원격 제어기는 마스터 스테이션으로 설정되어 있다. 그림의 위 부분은 유전 연산자(genetic operator)에 의하여 진화되는 PID 게인 조정기(PID tuner)로서, 다음과 같은 세 부분으로 구성되어 있다: a) 플랜트 출력(y)이나 최대오버슈트(Mp), 정착시간(Ts)과 같은 시스템의 스텝 응답 특성으로부터 스트링의 적합도를 계산하는 적합도 계산기(fitness calculator), b) 각 스트링의 적합도를 이용하여 자손(offspring)을 생성하는 GA 부분, c) 스트링을 PID 게인으로 변환하는 PID 계산기(PID calculator).

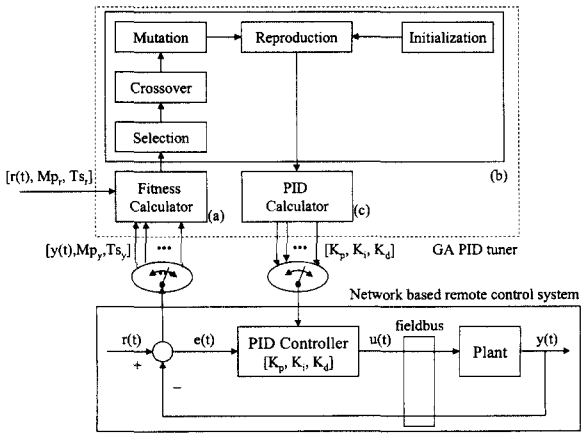


그림 7. GA에 의한 원격 제어기의 개념도.
Fig. 7. Schematic diagram of remote controller tuned by GA.

GA에 의한 원격 제어기의 튜닝 절차는 다음과 같다.

① 기준 입력(reference input, r), 기준 오버슈트(reference overshoot, Mp_r), 기준 정착시간(reference settling time, Ts_r)과 같은 플랜트의 설계사양을 선정하고, GA의 초기화 루틴을 통하여 인구(population) 수 만큼의 PID 게인 Kp, Ki, Kd를 구한다.

② 시스템에 기준 입력을 인가하고, Kp, Ki, Kd를 변화시켜 가면서 플랜트의 출력(y), 최대오버슈트(Mp_y), 정착시간(Ts_y)을 구한다.

③ 플랜트의 스텝 응답 특성과 설계사양(r, Mp_r, Ts_r)을 이용하여 적합도를 계산한다.

④ 여러 개의 서브스트링(sub-string)을 연결하여 하나의 스트링을 형성하는 다변수 코딩 방법(multi parameter coding) [13]을 이용하여, 3개의 PID 게인을 하나의 스트링으로 변환한다. 그리고 나서, 적합도와 유전 연산자(genetic operator)를 이용하여 새로운 스트링을 생성한다.

⑤ 생성된 스트링을 PID 게인으로 변환하고, GA의 종단조건(termination condition) 만족되거나 수행 횟수가 끝 날 때까지 2~5 단계를 반복한다.

2. 스트링의 구성

유전자 알고리즘에서는 PID 튜닝을 위하여 제어기의 Kp, Ki, Kd와 같은 PID 게인을 그대로 이용하지 않고, 스트링으로 변환하여 사용한다. 일반적으로, PID 게인은 이진수로 직접 코딩되며, PID 게인의 상한과 하한의 범위가 경험적으로 결정되는 경우가 대부분이다. 그러나, 이러한 코딩 방법은 PID 게인의 범위와 소수점 이하의 자릿수 같은 정밀도를 결정해야 하는 어려움이 존재한다[14]-[16].

이에 따라, 본 논문에서는 고전적인 제어기 설계 방법(classical controller design method)에 의하여 구한 PID 게인에 스케일링 계수(scaling factor)를 곱한 값을 제어기의 PID 게인으로 정의하고, 이 스케일링 계수를 GA의 스트링으로 선택하는 스케일링 계수 코딩 방법(scaling factor coding method)을 제안한다. 그림 8에는 스케일링 계수 코딩 방법에 대하여 나타내었으며, 그 절차는 다음과 같다;

① 원격 제어기의 PID 게인을 $Kp=\alpha Kp^T$, $Ki=\beta Ki^T$, $Kd=\gamma Kd^T$로 설정하고, 고전적인 제어기 설계 방법에 따라 Kp^T,

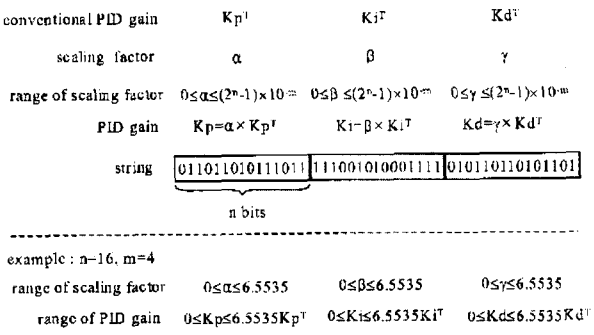


그림 8. 스케일링 계수 코딩 방법의 예시(n=16, m=4).
Fig. 8. Example of tuning factor coding method(n=16, m=4).

Ki^T, Kd^T를 구한다. 여기에서, 스케일링 계수 α, β, γ는 각각 0에서 $(2^n-1)\times10^{-m}$의 범위를 갖도록 적절한 스트링의 길이(length of string) n과 축적 계수(magnification factor) m을 선정한다.

② 스케일링 계수 α, β, γ를 GA의 스트링으로 선택하고, 4.1절에서 제시된 튜닝 절차에 따라 GA 연산을 거친다.

③ GA 연산을 거쳐 재생산된 스트링을 10진수로 변환한 후 10^m을 곱하여 α, β, γ를 계산한다. 그리고 나서, 이 값을 고전적 설계 방법에 의한 PID 게인에 곱하여 원격 제어기의 PID 게인 Kp, Ki, Kd를 계산한다.

이상과 같은 방법에 따라, 스트링의 길이 n과 축적 계수 m을 조절함으로써, PID 게인의 탐색 범위를 변화시키거나, 해상도를 쉽게 향상시킬 수가 있게 된다. 예로, n을 16으로, m을 4로 하게 되면, α, β, γ는 각각 0에서 $(2^{16}-1)\times10^{-4}$의 값(6.5535)을 가지게 되며, 소수점 4자리 수까지 표현할 수 있다.

3. 적합도 함수

GA에서 적합도 함수는 스트링의 우수성을 평가하기 위한 척도로서, 적자생존의 법칙에 따라 보다 높은 적합도를 가진 스트링이 더 많은 자손을 생성할 수 있는 확률이 높게 된다.

본 논문에서는 고전적인 설계 기준(classical performance criteria)과 최적 제어적 설계 기준(performance index criteria)을 결합시킨 적합도 함수를 제안한다. 일반적으로, 제어기 설계에 있어서, 근궤적법과 같은 고전 제어에서는 최대오버슈트이나 정착시간과 같은 플랜트의 설계사양을 이용하여 제어기를 설계한다[17]. 반면, 최적 제어에서는 ISE(integral of the square of the error)나 IAE(the integral of the absolute magnitude of the error)와 같은 성능 지수(performance index)를 만족시킬 수 있는 제어기를 설계한다[18].

본 논문에서는 (1)과 같이 가중치를 곱한 고전적 설계 기준과 성능 지수에 의한 설계 기준의 조합을 적합도 함수로 설정하였다.

$$\begin{aligned} f &= 1-(aF_1+bF_2) \\ &= 1-\{a[f_1(Mp_r, Mp_y)+f_1(Ts_r, Ts_y)]+bf_2(T)\} \end{aligned} \tag{1}$$

$$where \quad f_1(u,v)=\begin{cases} 0 & at\ v\le u \\ \dfrac{v-u}{u} & at\ u<v\le 2u \\ 1 & at\ v>2u \end{cases}$$

$$f_2(T)=\frac{\int_0^t (y(t)-r(t))^2\,dt}{\int_0^T r(t)^2\,dt}$$

식에서 첫 번째 항인 $f_1(Mp_r, Mp_y)$와 $f_1(Ts_r, Ts_y)$는 플랜트의 Mp_y, Ts_y와 설계사양 Mp_r, Ts_r로의 차이를 설계 사양으로 나눈 값으로서 0에서 1의 범위를 가지며, 두 번째 항인 $f_2(T)$는 정규화된 ISE로서 0에서 1의 범위를 가진다. 특히, 각 설계 기준에 대한 비중을 다르게 하기 위하여, 각 항에 가중치를 곱하였으며, 가중치들의 합은 1로 설정하였다. 이렇게 가중치를 조절함으로써, 두 개의 설계 기준 중 하나를 우선적으로 만족시킬 수가 있게 된다.

4. 유전 연산자

단순 GA(simple GA)에서는 선택(selection)과 교배(crossover), 돌연변이(mutation)와 같은 유전 연산자를 사용한다. 본 논문에서는 선택 방법으로서 stochastic remainder technique을 사용하였으며, 다수의 스트링의 적합도가 유사한 경우 부모와 동일한 비율의 자손들이 생성되는 것을 막기 위하여, (2)와 같은 선형 스케일링(linear scaling)을 사용하였다[12].

$$f' = pf + q \tag{2}$$

$$where \quad c=\frac{f_{max}-f_{min}}{f_{avg}-f_{min}}$$

$$p=\frac{(c-1)f_{avg}}{f_{max}-f_{avg}}$$

$$q=\frac{f_{avg}(f_{max}-cf_{avg})}{f_{max}-f_{avg}}$$

식에서 f는 적합도 함수이며, f'은 스케일링(scaling)된 적합도 함수이다. 또한, f_{max}는 적합도 함수의 최대값이며, f_{min}은 최소값, f_{avg}는 평균값이다.

교배 연산자로는 스트링들의 더 많은 탐색(perturbation)을 위하여 균일 교배(uniform crossover)를 사용하였으며, 돌연변이 연산자로는 한 점 돌연변이(one point mutation)을 사용하였다. 또한, 더 빠른 진화를 위하여, 현 세대의 최상의 개체가 다음 세대까지 살아 남을 수 있는 엘리티스트 모델(elitist model)을 사용하였다[12].

V. DC 모터를 이용한 성능 평가

본 절에서는 제안된 GA에 의한 원격 제어기의 성능을 평가하기 위하여, Profibus-DP를 이용한 네트워크 기반 제어 시스템의 실험 모델을 구현하고, 성능 평가를 수행하였다.

1. 플랜트 선정

본 절에서는 네트워크 기반 제어 시스템의 성능을 평가하기 위하여, Tamagawa사의 TS3728 DC 모터를 플랜트로 선정하였다. 플랜트의 제어 입력을 위하여 12비트 D/A 컨버터, 엔코더 출력을 위하여 16비트 카운터를 사용하였으며, 모터 샘플링 시간은 10ms로, 기준입력은 700rpm으로 설정하여 속도제어를 수행하였다.

스텝응답법을 이용하여 구한 DC 모터의 전달함수는 식 (3)과 같다[17].

$$G(s)=\frac{y(s)}{u(s)}=\frac{543.47}{0.12s+1} \quad (3)$$

여기에서, PID 제어기의 설계사양을 최대오버슈트는 10% 이하, 5% 정착시간은 0.8sec 이하, 감쇠비는 0.5 이하로 설정하고, 근궤적법(root locus)을 이용하여 PID 제어기를 설계하면 (4)와 같다[17].

$$G_{PID}(s)=K_d s+K_p+\frac{K_i}{s}=\frac{0.0001s^2+0.011s+0.2}{s} \quad (4)$$

2. 네트워크 기반 제어 시스템 실험 모델

그림 9는 Profibus-DP를 이용한 네트워크 기반 제어 시스템의 실험 모델을 나타내고 있다. 실험 모델에서, 1대의 제어기와 3대의 DC 모터가 Profibus-DP 네트워크에 연결되어 있으며, 1대의 제어기가 3대의 DC 모터를 동시에 제어한다. 이는 3축 로봇이나 가공기 등에서 중앙 제어기가 각 축의 구동기를 제어하는 것으로 간주할 수 있다.

실험 모델에서, 각 스테이션들의 네트워크 인터페이스로는 Siemens의 SAB-C165 마이크로컨트롤러와 마스터 및 슬레이브 전용 프로토콜 소자인 ASPC2와 SPC3을 장착한 Softing 사의 PROFI-board를 사용하였다. Profibus-DP의 전송속도는 1.5Mbps로 설정하였으며, 데이터 전송량은 2바이트(byte)로 설정하였다. 여기에서, 데이터 전송량이 2바이트인 이유는 DC 모터의 엔코더 카운트 신호가 16비트이므로 2바이트로 처리가 가능하기 때문이다. 또한, DC 모터의 샘플링 주기는 10msec로 설정하였으며, Profibus-DP의 폴링 주기는 각 프로세스들의 동작 주기나 플랜트의 샘플링 시간(T_{CYC})보다 4배 이상 빨라야 한다는 PNO의 권고[11]에 따라 2msec로 설정하였다.

그림 10은 네트워크없이 직접 연결된 제어시스템과 네트워크를 통하여 연결된 제어시스템의 응답 성능을 비교하고 있다. 그림에서, 직접 연결된 제어기의 응답 성능은 최대오버슈트는 7%, 정착시간은 0.73sec로서, 플랜트의 설계사양을 만족시키고 있음을 알 수 있다. 그러나, 네트워크를 통하여 연결된 제어기의 경우, 최대오버슈트는 16%, 정착시간은 1.12초로 설계사양을 초과하였다. 이러한 결과는 모터의 피드백 신호가 네트워크 지연으로 인하여, 3~5 샘플링 주기정도 늦게 제어기에 도착하기 때문에 발생한다. 따라서, 제어시스템의 설계사양을 만족시키기 위해서는 네트워크 지연을 고려한 원격 제어기 설계가 이루어져야 함을 알 수 있다.

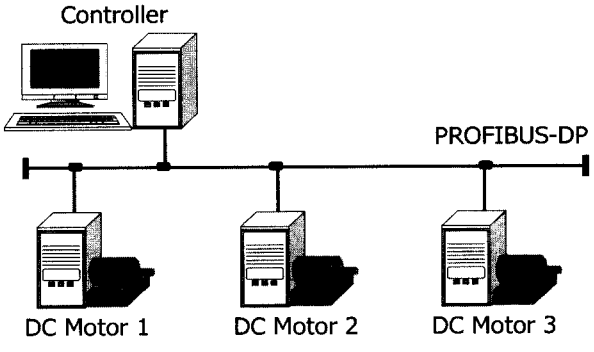


그림 9. 네트워크 기반 제어 시스템의 테스트베드.
Fig. 9. Network testbed for networked control system.

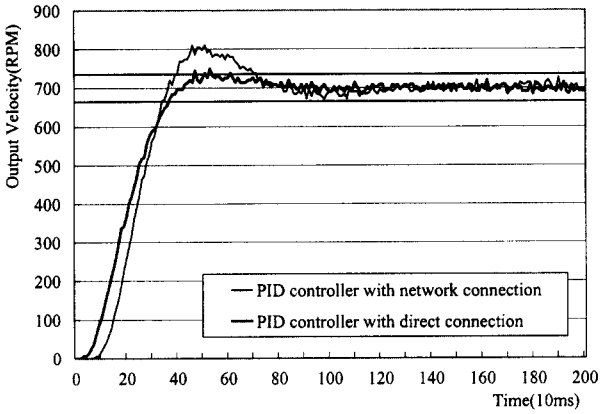


그림 10. NCS에서 네트워크 지연의 영향.
Fig. 10. Effect of network delay on the NCS.

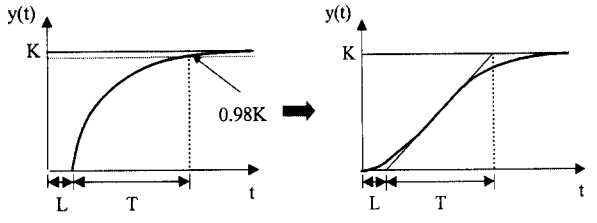
3. GA에 의한 원격 제어기의 성능 평가

본 절에서는 제안된 원격 제어기의 성능의 우수성을 평가하기 위하여, 고전적인 방법에 의하여 설계된 PID 제어기의 성능과 비교하였다.

고전적인 방법을 적용하기 위하여, 본 절에서는 그림 11과 같이 네트워크 지연 L을 공정 시스템의 시간 지연 L로서 가정하였으며, 가장 널리 사용되고 있는 Ziegler-Nichols 계수조정법[19]을 사용하여 PID 제어기를 설계하였다. 여기에서, Ziegler-Nichols 계수조정법에서 시간 지연은 3 샘플링 타임, 즉 30msec가 발생하는 것으로 가정하였다.

본 절에서 사용된 GA의 파라미터와 적합도 함수의 파라미터들은 표 1과 같다. 먼저, GA의 개체 수는 30으로 하였으며, 교배 확률은 0.9, 돌연변이 확률은 0.01로 설정하였다. 또한, 스트링의 길이 n은 각각 16비트로 설정하였으며, 축적계수 m은 4로 설정하였다. 적합도 함수의 경우, 고전적 설계 기준 $f_1(u,v)$의 Mp_r는 7%, Ts_r는 0.55sec로 설정하였으며, 성능지수에 의한 설계 기준 $f_2(T)$의 적분 구간은 0에서 2초로 설정하였다. 또한, 가중치 a는 0.4로, b는 0.2로 설정하였다.

그림 12a는 100세대가 진화되는 경우의 적합도를 나타내며, 그림 12b는 고전적인 설계 기준과 성능 지수에 의한 설계 기준의 적합도를 나타낸다. 그림에서 적합도는 초기에 대략 0.75822에서 점차적으로 향상되어 53세대가 지난 후에



(a) step response with network delay (b) step response with time delay

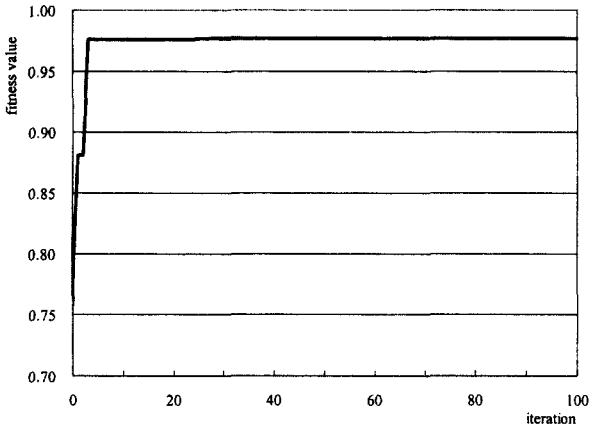
그림 11. 네트워크 지연과 시간 지연의 비교.
Fig. 11. Comparison of network delay and time delay.

표 1. GA와 적합도함수의 파라미터.
Table 1. Parameters for GA and fitness function.

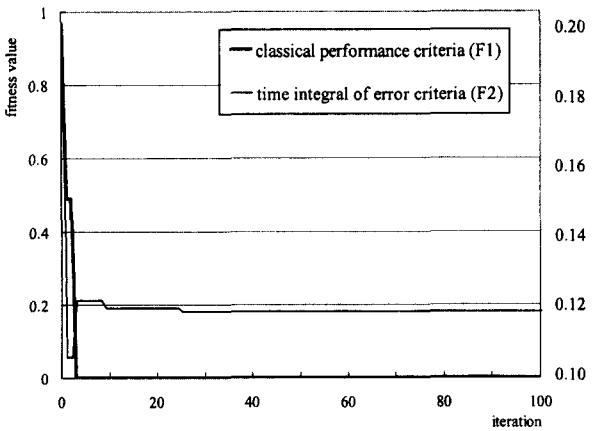
Population size	30
Crossover probability	0.9
Mutation probability	0.01
String length (bit)	16
Magnification factor	4
Mp_r	7%
Ts_r	0.55sec
T	2sec
a	0.4
b	0.2

는 0.97643으로 수렴하였다. 특히, 오버슈트와 정착시간에 대한 적합도는 3세대에서 이미 만족되고, 그 이후의 적합도 향상은 ISE의 최소화에 의하여 이루어짐을 확인할 수 있었다. 100세대가 지난 후의 α, β, γ은 1.2098, 0.6886, 0.6160으로 나타났으며, 이를 이용하여 원격 제어기의 게인을 계산하면, Kp=0.0133, Ki=0.1377, Kd=0.0000616으로 나타났다.

그림 13과 표 2는 Ziegler-Nichols 계수조정법에 의하여 튜닝된 PID 제어기(Kp=0.008832, Ki=0.147, Kd=0.000132)에 의한 응답과 100세대 진화를 통하여 조절된 원격 제어기에



(a) fitness function



(b) classical performance criteria and performance index criteria

그림 12. 게인 튜닝동안의 적합도 함수의 경향.
Fig. 12. Trend of fitness values during the gain tuning.

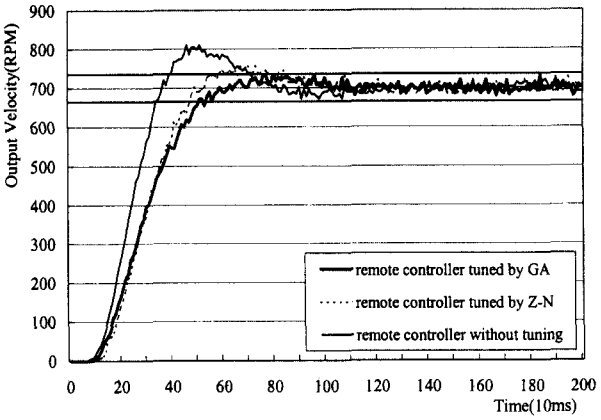


그림 13. Ziegler-Nichols법과 GA에 의해 튜닝된 원격 제어기의 응답 비교.
Fig. 13. Comparison of responses of remote controllers tuned by GA and Ziegler-Nichols methods.

표 2. PID 게인과 응답성능 비교.
Table 2. Comparison of PID gains and control system Performance.

Method	Ziegler-Nichols	GA
Kp	0.008832	0.0133
Ki	0.147	0.1377
Kd	0.000132	0.0000616
Mp	8%	4.57%
Ts	0.85sec	0.54sec

의한 응답을 비교하였다. 그림에서 Ziegler-Nichols 계수조정법에 의하여 튜닝된 PID 제어기를 이용한 결과, 최대오버슈트는 8%로, 정착시간은 0.85초로 나타났다. 반면, GA에 의하여 조절된 원격 제어기를 이용한 결과, 최대오버슈트는 4.57%로, 정착시간은 0.54초로 나타났다. 이러한 결과로 볼 때, GA에 의한 방법이 Ziegler-Nichols 방법에 비해 성능이 우수함을 알 수 있었다. 또한, Ziegler-Nichols 방법은 네트워크 지연을 직접 측정하여 그 크기를 알 필요가 있지만, GA에 의한 방법은 네트워크 지연에 대한 정보를 알 필요가 없어도 효과적인 튜닝이 이루어짐을 알 수 있다. 특히, 네트워크 지연은 시스템의 구현 방식이나 네트워크의 통신 부하, 접속되는 스테이션의 수 등에 따라 변하므로, 시스템의 성능 평가를 충분히 수행하기 전까지는 정확한 값을 알기는 매우 어렵다. 따라서, 불확실한 네트워크 지연을 가지는 네트워크 기반 제어 시스템에서 제어기 설계 방법으로는 GA에 의한 제어기 설계 방법이 더 우수함을 알 수 있다.

VI. 결론

본 논문에서는 Profibus-DP를 이용한 네트워크 기반 제어 시스템의 구축 방법을 제안하고, 네트워크 지연의 발생 원인을 규명하였다. 또한, GA에 의한 네트워크 기반 제어기의 설계 방법에 대하여 제안하였다. 이러한 결과로서 다음과 같은 결론을 얻을 수 있었다.

첫째, 네트워크 기반 제어 시스템에서 네트워크 지연이

발생함을 관찰하였으며, 이러한 지연이 제어 시스템의 전체적인 성능을 저하시킨다는 점을 실험적으로 확인하였다. 따라서, 필드버스를 이용한 네트워크 기반 제어 시스템의 설계에 있어서 네트워크 지연을 고려한 제어기 설계가 이루어져야 함을 확인하였다.

둘째, Profibus-DP에서의 네트워크 지연은 프로토콜의 특성과 각 프로세스들간의 비동기화로 인하여 발생함을 확인하였다. 특히, 응용 프로세스에서의 비동기화는 제어기와 플랜트의 응용 프로세스 설계 방식에 따라 증가될 수 있으므로, 네트워크 지연을 최소화시킬 수 있는 방향으로 설계되어야 한다.

셋째, 유전자 알고리즘을 이용하여 PID 제어기를 기초로 한 원격 제어기를 설계하는 방법은 네트워크 지연에 대한 직접적인 정보가 없는 상황에서도 효과적임을 확인하였다.

참고문헌

[1] A. Ray, "Networking for computer-integrated manufacturing," *IEEE Network*, vol. 2, no. 3, pp. 40-47, 1988.

[2] 홍승호, "필드버스 기술 동향," 제어 · 자동화 · 시스템공학회지, 제4권, 제6호, pp. 13-18, 1998.

[3] 권욱현, 김영신, "분산 실시간 시스템에서의 네트워크 프로토콜," 제어 · 자동화 · 시스템공학회지, 제4권, 제6호, pp. 27-34, 1998.

[4] 이경창, 김기웅, 김희현, 이석, "실시간 페루프 제어 시스템을 위한 Profibus-FMS 네트워크의 구현," 제어 · 자동화 · 시스템공학회 논문지, 제6권, 제5호, pp. 92-98, 2000.

[5] IEC 61158-4, *Digital data communications for measurement and control - Fieldbus for use in industrial control systems - Part 4: Data link protocol specification*, IEC, 1999.

[6] S. Cavalieri, A. D. Stefano, and O. Mirabella, "Impact of fieldbus on communication in robotic systems," *IEEE Transactions on Robotics and Automation*, vol. 13, no. 1, pp. 20-48, 1997.

[7] G. C. Walsh, and Y. Hong, "Scheduling of networked control systems," *IEEE Control Systems Magazine*, vol. 21, no. 1, pp. 57-65, 2001.

[8] Z. Wei, M. S. Branicky, and S. M. Phillips, "Stability of networked control systems," *IEEE Control Systems Magazine*, vol. 21, no. 1, pp. 84-99, 2001.

[9] EN 50170, *Profibus Specification - Normative Parts of Profibus-FMS, -DP, -PA according to the European Standard*, vol. 2, 1998.

[10] *Profibus Application Program Interface, user manual, version 5.2, rev.01*, 1998.

[11] PNO, *Profile for Variable Speed Drives, PROFIDRIVE, version 2.0*, Profibus International, 1997.

[12] D. E. Goldberg, *Genetic Algorithms in Search, Optimization and Machine Learning*, Addison-Wesley, 1989.

[13] 이상호, 김인준, 이경창, 이석, "공장자동화용 네트워크를 위한 통합 성능관리기의 개발," 제어 · 자동화 · 시스템공학회 논문지, 제5권, 제5호, pp. 600-613, 1999.

[14] B. S. Chen and Y. M. Cheng, "A structure-specified H_∞ optimal control design for practical applications: a genetic approach," *IEEE Transactions on Control System Technology*, vol. 6, no. 6, pp. 707-718, 1998.

[15] D. A. Scott, C. L. Karr, and D. E. Schinstock, "Genetic algorithm frequency-domain optimization of an anti-resonant electromechanical controller," *Engineering Applications of Artificial Intelligence*, vol. 12, no. 2, pp. 201-211, 1999.

[16] Y. Mitsukura, T. Yamamoto, and M. Kaneda, "A design of self-tuning pid controllers using a genetic algorithms," *Proceedings of the American Control Conference*, pp. 1361-1365, 1999.

[17] K. Ogata, *Modern Control Engineering*, Prentice-Hall, pp. 257-270, 1990.

[18] R. C. Dorf and R. H. Bishop, *Modern Control Systems*, Addison-Wesley, 1998.

[19] J. G. Ziegler and N. B. Nichols, "Optimum settings for automatic controllers," *Transactions on ASME*, vol. 64, 1942.

이 경 창

1971년 5월 1일생. 1996년 부산대 생산기계공학과 졸업. 동대학원 석사(1998년). 동대학원 박사수료(2001년). 1998년~현재 기계공학연구정보센터 전임연구원. 관심분야는 자동화용 네트워크(Profibus, Fieldbus Foundation, Lonworks)의 설계 및 성능 평가, 지능 제어, 인터넷 기반 생산 시스템.

이 석

1961년 12월 11일생. 1984년 서울대 기계공학과 졸업. 펜실바니아 주립대 석사(1985). 동대학원 박사(1990년). 1990년~1993년 신시내티 대학교 기계공학과 조교수. 1993년~현재 부산대학교 기계공학부 부교수. 관심분야는 자동화용 네트워크(Prifobus, Fieldbus Foundation, Lonworks)의 설계 및 성능 평가, 차량용 네트워크(CAN, LIN, KWP2000)의 설계 및 성능 평가, DES, RP, 자율주행.