메시지 지연시간을 고려한 CAN 기반 피드백 제어시스템의 응답특성 분석

論 文
51D-5-3

Analysis of Response Characteristics of CAN-Based Feedback Control System Considering Message Time Delays

全 鐘 萬* · 金 大 元**
(Jong-Man Jeon · Dae-Won Kim)

Abstract - In this paper, the response characteristics of CAN-based feedback control system are analyzed when message time delays through the network are considered. The message time delays are composed of computation time delay and communication time delay. The application layer of CAN communication is modeled mathematically to analyze two time delays, and the communication time delay is redefined under several assumption conditions. The CAN-based feedback control system is proposed as a target system that is the machining system with the three axes. The response characteristics of time delays in the proposed system are analyzed through computer simulations, and can be improved by the compensation using the PID tuning method to satisfy the design specifications of the system.

Key Words : Time delays, Real-time control system, CAN protocol, PID tuning

1. 서 론

중앙집중형 제어시스템은 시스템의 구조가 복잡해지고 다양해지면서 많은 양의 데이터 처리 요구에 부합하지 못하는 여러 가지 단점들, 예컨대, 많은 배선의 필요, 시스템의 확장 및 유지 보수의 어려움 등을 가지고 있다. 이러한 문제로 인하여 중앙집중형 연결방식의 시스템들은 네트워크를 이용한 분산형 연결방식의 분산 실시간 제어시스템으로 대체되고 있는 추세이다. 네트워크 기반 분산 실시간 제어시스템은 시스템 재구성과 배선작업이 용이하며 비용이 적게 드는 장점들을 가지고 있기 때문에 현재 여러 산업 자동화 분야에 응용되고 있다[1,2,3].

네트워크 기반 분산 실시간 제어시스템을 구성하는데 있어서 네트워크 프로토콜은 실시간 제어를 가능하게 하는 중요한 요소로서 작용한다. 제어 네트워크 프로토콜을 실시간 시스템에 적용하기 위해서는 프로토콜의 실시간 특성뿐만 아니라 스케줄성 분석을 통해 시스템에 적용 가능한지를 고려하여 적합한 제어 네트워크 프로토콜을 선정해야 한다[4].

1980년대 BOCSH사와 Intel사에서 자동차 시스템에 적용하고자 개발된 CAN(Controller Area Network) 프로토콜은 고속의 통신 인터페이스를 제공하고 데이터 프레임의 오버헤드(overhead)가 적기 때문에 빠른 응답특성을 갖고 있다. 또한, 식별자(identifier)를 이용한 충돌방지와 전송중재(arbitration) 기능을 갖고 있어 실시간 제어 네트워크 프로토콜로서 피드백 제어를 요구하는 시스템 환경에 적합하다[5].

네트워크를 통해 데이터를 전송할 때, 메시지 초기화와 메시지 전달을 위해 걸리는 지연시간은 네트워크의 속도나 메시지 크기, 네트워크의 트래픽, 그리고 사용된 프로토콜의 영향을 받는다. 또한, 통신 프로토콜은 네트워크 기반 제어시스템의 신뢰성과 오차허용(fault-tolerance)에도 영향을 준다. 네트워크를 기반으로 하는 피드백 제어시스템 구성시 필연적으로 발생하는 지연시간은 시스템을 불안정하게 하는 요소로서 작용될 수 있다[6]. 따라서 지연시간 발생 요소의 분석을 통해 제어기의 응답특성을 고려하여 제어시스템을 설계하여야 한다.

통신상의 지연시간에 대한 분석은 기존의 많은 연구에서 진행되어 왔다. Tindell과 Burns[8,9]는 통신 종단간(end-to-end com-munication) 메시지 전송에 있어 발생하는 지연시간 요소를 네 가지로 정의하였고, 지연시간의 수학적 모델링을 통해 실시간 스케줄성 분석을 수행하였다. 이러한 수학적 모델을 CAN 프로토콜에 적용하여 통신상에 발생하는 지연시간에 대한 분석을 수행하였다[10,11]. 또한, 메시지의 최악의 응답시간을 계산해 봄으로써 실시간 성능 분석에 적용해 보았다. 그리고 로컬 영역에서의 메시지 발생으로 인한 발생 지연시간에 대한 수학적 모델링을 통해 CAN 통신상에 발생할 수 있는 모든 지연시간 요소에 대한 분석을 수행하였다[12]. 그러나, 이러한 연구 결과들은 메시지 스케줄링에 따른 실시간 분석을 통해 제어 가능성을 판단하고 시스템에 적용이 가능한지를 평가할 수 있는 방법만을 제시하였다. 따라서, 피드백 제어 환경에서는 제어기의 지연특성을 시간 지연의 특성으로 고려하여 제어기의 응답 특성을 만족하게 하기 위해서는 지연시간에 대한 구체적이고 정량적인 분석 결과가 요구되어진다. 또한, 이러한 분석 결과를 통해

* 準 會 員 : 明知大 工大 情報制御工學科 碩師課程
** 正 會 員 : 明知大 工大 情報工學科 敎授
接受日字 : 2001年 11月 30日
最終完了 : 2002年 3月 15日

제어기의 응답특성을 보상함으로써 시스템의 응답특성을 만족할 수 있도록 설계할 수 있다.

본 논문에서는 CAN 통신상의 지연시간을 계산지연시간(computation time delay)과 통신지연시간(communication time delay)으로 정의하고, 지연시간에 대한 구체적이고 정량적인 분석을 수행한다. 그리고 분석된 결과를 이용해 통신지연시간 요소인 미디어 접근지연시간과 전송지연시간의 관계를 통해 통신지연시간의 수학적 모델을 유도한다. 또한, 분석된 지연시간을 제어기의 시간지연 특성으로 고려하여, 시간지연이 있는 시스템으로 재모델링한다. 그리고 PID 계수 조정법을 이용해 제어시스템의 시간지연 응답특성을 계수 조정을 통해 보상한다.

본 논문의 구성을 살펴보면, 제 2장에서는 CAN 프로토콜의 특징 및 실시간 특성에 대해 간략히 서술하고 3장에서는 CAN 통신 응용계층의 모델링과 지연시간에 대한 구체적이고 정량적인 분석을 수행하고 4장에서는 CAN 기반 피드백 제어시스템의 구성과 지연시간을 고려한 제어기의 응답특성을 모의실험 결과를 통해 알아본다. 마지막으로 5장에서는 결론을 서술한다.

2. CAN 프로토콜의 특성

CAN 통신 프로토콜은 디바이스들간의 정보 교환 방식을 ISO의 OSI 참조 모델에 의거하여 7개의 계층 중에 하위 2개층인 데이터링크 계층과 물리 계층을 사용하여 정의하고 있다[7]. 이더넷이 CSMA/CD 방식을 사용하는 것과는 달리, CAN 프로토콜은 데이터 링크 계층에서의 미디어 접근을 위해 충돌을 감지하고 중재할 수 있는 CSMA/AMP(Carrier Sense Multiple Access/Arbitration on Message Priority) 방식을 사용하고 있다. CAN 통신 프로토콜의 가장 큰 장점 중에 하나가 고유한 식별자를 통해 메시지에 우선순위를 부여할 수 있다는 것인데, 이 식별자를 통해 메시지간의 충돌을 방지하고 중재 역할을 한다. 메시지에 대한 식별자 부여는 사전 스케줄링이나 시스템 운영시에 동적으로 부여할 수 있다.

CAN 통신 프로토콜의 실시간 특징은 다음과 같다. 첫째, 고속의 직렬 통신 인터페이스로서 최대 1Mbps까지의 전송속도를 가진다. 둘째, 최소 0 바이트에서 최대 8 바이트까지의 짧은 메시지 길이는 많은 양의 데이터 전송을 요구하지 않는 실시간 시스템에 알맞은 특징중의 하나이다. 셋째, PROFIBUS-DP 프로토콜이 토큰을 얻어 메시지 전송에 대한 허가를 얻는 것과는 달리[15], 토큰이나 어떠한 허가 없이도 전송이 가능하기 때문에 빠른 반응시간의 특징이 있다. 마지막으로, 에러 자동 검출 기능이 있어 신뢰성이 높다. 따라서, 이러한 CAN의 실시간 특징은 분산 실시간 네트워크 기반 제어시스템의 제어 네트워크 프로토콜로서 적합하다.

3. CAN 통신 응용계층 모델 및 지연시간 분석

CAN 통신 지연시간은 계산지연시간과 통신지연시간으로 크게 두 가지로 구분할 수 있다. 본 장에서는 CAN 통신 응용계층을 모델링하고, 지연시간 발생 요소별로 구체적이고 정량적인 분석을 수행한다. 또한, 로컬 노드에서의 지연시간을 상수로 고려하고 몇 가지 제한 조건하에서 미디어 접근지연시간과 전송지연시간을 수학적으로 재정의한다.

3.1 CAN 인터페이스 구조

CAN 인터페이스는 구현 특징상 일반적으로 3가지 형태로 구분할 수 있다. 이러한 형태로는 독립 CAN 컨트롤러(stand alone CAN controller), 통합 CAN 컨트롤러(integrated CAN controller),
그리고 단일 칩 CAN 컨트롤러(single-chip CAN contoller)로 구분된다[14]. 다음 그림 1은 각각의 컨트롤러의 형태별 구조를 나타낸다.

통합 CAN 컨트롤러(그림1 (b))는 마이크로 컨트롤러(micro-controller)와 CAN 컨트롤러가 하나의 회로 내에 구성되기 때문에 독립 CAN 컨트롤러(그림1 (a))에 비해 가격이 싸고 공간 요구가 적다는 장점이 있지만 CPU에 의존적이므로 재사용성에 있어서 다소 떨어지는 단점이 있다. 그러나 단일 칩 컨트롤러와 비교해 볼 때 재사용에 있어 더욱 융통적이어서 현재 많이 사용되고 있는 CAN 인터페이스 구조이다.

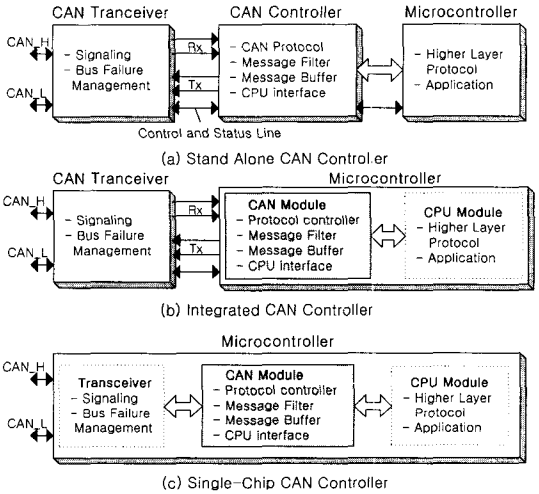

그림 1 구현 특징에 따른 CAN 인터페이스 구조
Fig. 1 An architecture of CAN interfaces

3.2 CAN 통신 응용계층 모델링

그림 2는 통합 CAN 컨트롤러를 이용한 CAN 인터페이스의 구조로 구성된 CAN 기반 제어시스템의 통신 응용계층의 모델을 나타낸다.

두 노드간에 발생하는 지연시간은 크게 계산지연시간과 통신지연시간으로 구분할 수 있다. 계산지연시간은 로컬 노드에서의 CAN 컨트롤러, 또는 트랜시버와 같은 하드웨어의 수행으로 인한 하드웨어적인 지연요소와 메시지 입출력, 또는 제어프로세스의 처리를 위한 소프트웨어적인 지연요소로 구성된다. 또한, 통신지연시간은 메시지가 다른 노드에 전송되기 위해서, 미디어를 점유하기 위해 지연되는 미디어 접근 지연요소와 메시지가 네트워크 미디어를 통해 전송되는 전송 지연요소로 이루어진다.

그림 2에서 보면, 독립 CAN 노드로부터 전송된 메시지는 트랜시버를 통해 CAN 컨트롤러에 전달이 된다. 수신된 메시지는 우선 통신 관리(communication management) 서비스에 의해 메모리에 저장되고 식별자를 이용한 메시지 필터링(message filtering)을 통해 응용 프로그램에 전달된다. 그리

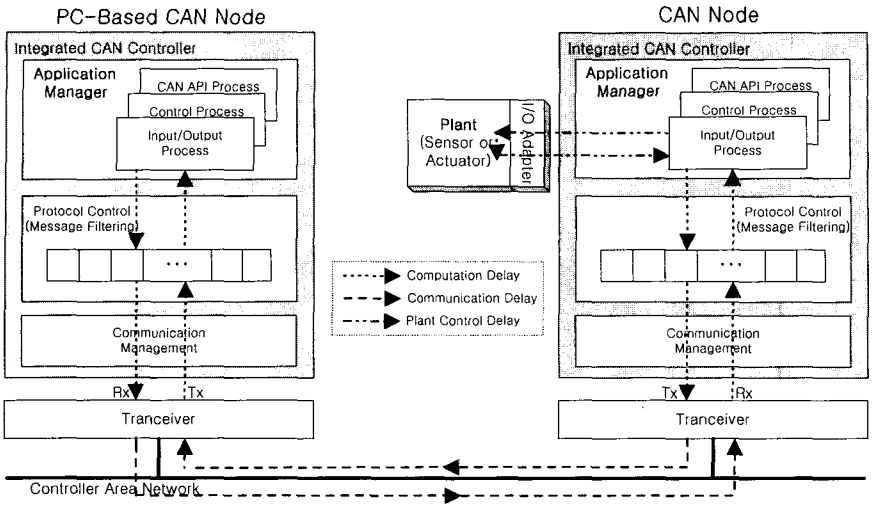

그림 2 CAN 통신 응용계층 모델

Fig. 2 An application model of CAN communication

고 응용 관리자(application manager)에 의한 제어프로세스가 수행된다. 반대로 송신 메시지의 처리는 응용 프로그램의 CAN API 프로세스에 의해 메시지 필터링을 한 후 통신 관리 서비스에 의해 트랜시버에 전달이 된다. 그리고 네트워크 미디어를 통해 다른 노드에 전달된다. 오른쪽의 독립 CAN 노드에는 센서나 액츄에이터(actuator)가 하드 커플링(hard coupling)되어 있으며 플랜트를 제어하기 위한 정보 교환이 이루어진다.

3.3 CAN 통신 지연시간

3.3.1 계산지연시간

계산지연시간은 소프트웨어 지연시간과 하드웨어 지연시간으로 구성된다. 소프트웨어 지연시간은 마이크로 프로세서에 따라 CPU 클럭의 명령어 수행시간이 다르기 때문에, 피드백 제어 알고리즘을 처리하기 위한 제어 프로세스의 수행과 CAN 규약 정의에 따른 프로세스를 처리하기 위한 CAN API 수행, 그리고 메시지 전송과 수신 처리를 위한 ISR(Interrupt Service Routine) 수행시간은 프로세서의 CPU가 한 클럭에 수행할 수 있는 명령어 처리 능력에 의존한다.

만일 80C196을 이용한 통합 CAN controller의 CPU 버스 주파수가 16MHz라고 가정한다면, CAN API의 수행시간은 다음과 같다: 메시지 수신을 위한 ISR 수행시간(38 μs ~92 μs), 메시지 전송을 위한 ISR 수행시간(17 μs ~74 μs), 메시지 메모리 로드(36 μs), 메시지 메모리에 읽기와 쓰기(39 μs), 그리고 메시지 전송을 위해 버퍼에 저장할 때 걸리는 시간(58 μs)으로 정의할 수 있으며 대략적으로 총 수행시간은 188.5 μs에서 299 μs까지의 시간이 걸린다. ISR 수행시간은 식별자의 메모리 저장 위치에 따라 접근 시간이 다르기 때문에 유동적으로 변화한다. 이러한 시간은 응용계층 제작자에 따라 다른 시간이 소요될 수 있다.

하드웨어 지연시간은 CAN 컨트롤러(CAN controller)와 옵토커플러(optocoupler), 트랜시버(transceiverr), 그리고 케이블(cable)에 의해 걸리는 시간을 말한다. 하드웨어적인 지연시간은 하드웨어 제작자에 의존하기 때문에 조금씩 차이가 있다. 최근에 제작되고 있는 CAN 인터페이스의 하드웨어 지연시간은 아주 작은 시간이 소요된다. CiA의 자료[14]에 의하면, 일반적으로 CAN 컨트롤러에 의한 지연시간은 50ns~62ns, 옵토커플러에 의한 지연시간은 40ns~140ns, 트랜시버에 의한 지연시간은 120ns~250ns, 그리고 케이블에 의한 지연시간은 대략 5ns/m으로 나타났다.

이러한 지연시간은 CAN의 특성상 물리적인 동기화(hard synchronization)가 선행되기 때문에 2배의 지연시간이 고려되어야 한다. 따라서 (1)과 같이 정의될 수 있으며 하드웨어적으로 지연되는 시간은 일반적으로 네트워크 미디어를 통해 1비트의 데이터를 전송하는 시간보다는 적게 소요되도록 제작된다. 다시 말하면, CAN 버스의 전송속도가 1Mbps일 때 1비트를 전송하는데 필요한 시간은 1 μs이므로 하드웨어 지연시간은 1 μs보다 작다.

$$t_{prop} = 2(t_{cab} + t_{cont} + t_{opto} + t_{tran}) \quad (1)$$

계산지연시간은 CAN 인터페이스와 응용프로그램의 제작자에 크게 의존하기 때문에 약간의 차이는 보일 수 있지만 상수로 가정할 수 있다.

3.3.2 통신지연시간

통신지연시간은 미디어 접근 지연시간과 전송지연시간으로 이루어진다. 미디어 접근 지연시간은 전송지연시간과 밀접한 관련이 있는데 메시지가 미디어를 점유하기 위해 대기하는 시간이므로 상위 우선순위를 갖는 메시지가 전송되는 시간만큼 지연된다.

본 논문에서의 제어 대상 시스템의 구성에 있어서, 각각의 노드는 하나의 고유한 식별자를 갖는 단일 메시지를 발생시키고 제어주기가 같은 피드백 제어를 위한 시스템이라고 가정한다. 또한, 각 노드의 메시지들은 동일한 순간에 발생된다고 하면, 다음 그림 3에서의 맨 아래의 굵은 부분처럼 시스템으로부터 발생하는 모든 메시지의 실행시간은 직렬화

(serialization)가 이루어지면서 실행이 된다. 이것은 메시지들이 우선순위에 따라 수행이 되기 때문에 우선순위가 가장 높은 메시지를 제외하고는 상위 우선순위의 메시지 실행만큼 지연이 되어 수행된다는 것을 의미한다. 또한, 기존의 미디어 접근 지연시간의 정의에서는 낮은 우선순위를 갖는 메시지가 미디어를 점유하므로 인해 발생되는 블록킹 시간(blocking time)을 고려했지만[8], 위의 가정 조건하에서는 고려되지 않는다.

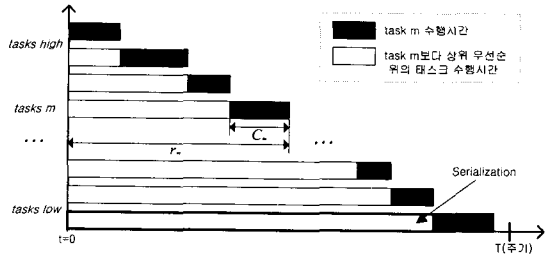

그림 3 메시지 수행 시간
Fig. 3 Execution time of messages

따라서 미디어 접근 지연시간은 상위 우선순위를 갖는 메시지의 미디어 점유로 인해 지연되는 시간만을 고려하면 된다.

전송지연시간의 수학적 모델은 다음과 같다[10].

$$C_m = \left(\left\lfloor \frac{34+8s_m}{5} \right\rfloor + 47 + 8s_m \right) \tau_{bit} \tag{2}$$

여기서, C_m은 전송지연시간을, s_m은 메시지 크기를, 그리고 τ_{bit}는 1비트를 전송하는데 걸리는 시간을 나타낸다. 또한, 오른쪽의 첫째항은 에러 검출을 위해 연속적으로 5개의 같은 비트(0 또는 1)가 전송될 때 극성이 다른 비트를 추가시키므로써 생기는 오버헤드(overhead)이고 이것을 비트 스터핑(bit stuffing)이라 한다[10]. 그리고, 47은 CAN 데이터 프레임의 오버헤드이다.

통신지연시간은 미디어 접근 지연시간이 전송지연시간의 영향으로 인해 발생하기 때문에 (3)과 같이 정의 될 수 있다.

$$t_{commd} = \sum_{\forall m \in hp(l)} C_m \tag{3}$$

여기서, t_{commd}는 메시지 전송을 위해 걸리는 통신지연시간을, $\forall m$은 메시지 집합을, 그리고 $hp(l)$는 메시지 m보다 높은 우선순위를 갖는 메시지의 집합을 의미한다. 통신지연시간은 각 노드에서 발생하는 메시지의 크기와 전송속도에 따라 차이를 보일 수 있다. CAN 버스의 전송속도는 최대 1Mbps까지 가능하고 거리에 반비례한다.

만일 메시지의 크기가 같다고 가정하면, 다음 (4)와 같이 재정의할 수 있다.

$$t_{commd} = C_m \times P_m \tag{4}$$

여기서, P_m은 메시지 우선순위를 나타내며 1~2032까지 정수의 값을 갖는다. 각 노드에서 동일한 시점에 발생한 메시지들은 CAN 데이터 프레임의 식별자 필드에 의해 결정된 우선순위에 따라 미디어를 접근하기 때문에 통신지연시간은 상위 우선순위가 미디어를 점유하는 시간만큼 지연된다. 따라서, 전송지연시간과 메시지의 우선순위의 곱으로 표현될 수 있다.

메시지 처리를 위한 피드백 제어 루프의 지연시간은 (5)와 같이 정의 될 수 있다.

$$t_{fcl} = 2 \times (t_{comp} + t_{prop} + t_{commd}) + t_{cp} \tag{5}$$

여기서, t_{fcl}은 피드백 제어 실행시간을, t_{comp}은 소프트웨어에 의한 계산지연시간을, t_{prop}은 하드웨어적인 요소에 의한 계산지연시간을, t_{commd}는 통신지연시간을, 그리고 t_{cp}는 제어 프로세스를 처리하는데 걸리는 시간을 나타낸다. 위의 식은 일반적인 식으로 정의되었지만, 정확히 정의 하자면 통신지연시간인 t_{commd}는 메시지의 출발 노드와 목적 노드의 우선순위에 따라 지연시간이 달라지기 때문에 각 노드의 우선순위를 고려하여 지연시간을 계산해야 하며 메시지에 대한 우선순위 할당은 사전 스케줄링 방법에 따라 정해진다.

4. 모의실험 및 결과 분석

4.1 CAN 기반 피드백 제어시스템의 구조

네트워크를 기반으로 하는 제어시스템에서 메시지 전송으로 인해 발생되는 지연시간은 제어시스템의 구조에 따라 다르게 나타난다[13]. 서보기능(servo-mechanism)을 하는 부분과 모터를 독립 노드로 구성하는 것 보다는 서보기능을 하는 모터를 사용하여 하나의 노드로 구성하는 제어시스템의 구조가 지연시간을 더 최소화할 수 있다.

본 논문에서의 CAN 기반 피드백 제어시스템의 구성요소는 주 컨트롤러부(main controller), 센서부, 구동부로 이루어진다. 주 컨트롤러는 센서로부터 들어오는 피드백 정보와 레퍼런스 값을 비교하여 에러값 보정을 위한 제어 프로세스를 처리하는 역할을 담당한다. 구동부와 센서부는 직선운동(linear slide)을 하는 축을 구동하기 위한 DC 모터와 위치정보 측정을 위한 앤코더로 구성된다. 또한, DC 모터는 속도정보를 제공하는 타코메타를 갖고 있다. 그림 5는 CAN 기반 제어시스템의 네트워크 구조를 보여준다.

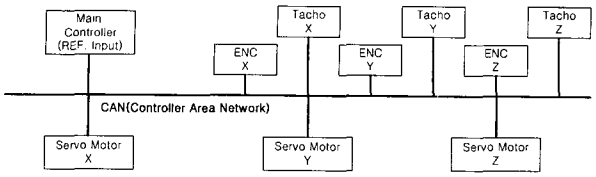

그림 5 네트워크 기반 제어시스템의 구조
Fig. 5 Architecture of network-based control system

네트워크 기반 피드백 제어시스템 구성시 메시지 지연시간의 발생 위치는 센서 정보의 입력과 제어 명령을 액츄에이터에 전송하는데 있어 존재할 수 있다. 또한 주 컨트롤러가 센서로부터 정보를 얻어 제어 프로세스를 수행하는데 있어 계산지연시간이 존재한다. 그림 6은 플랜트와 주 컨트롤러 사이의 피드백 제어 루프와 지연시간 발생 위치를 보여준다.

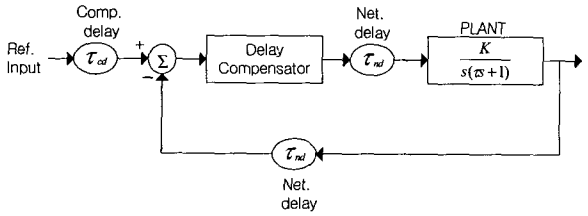

그림 6 메시지 피드백 제어 루프
Fig. 6 Message feedback control loop

제어기의 응답에 있어서 발생하는 지연시간을 시간지연의 개념으로 고려하여 시간지연 함수를 플랜트 전달함수에 추가시키고 PID 제어기의 계수값을 구하는 방식으로서 Ziegler-Nichols의 전통적인 PID 계수 조정법을 이용할 수 있다. 그리고 PID 계수는 표 1과 같이 구해진다. 또한, 지연요소를 갖는 시스템의 응답특성이 그림 7과 같이 S자 모양으로 나타나면, (6)과 같이 간단하게 1차식의 전달함수로 표현할 수 있다[6].

$$\frac{Y(s)}{U(s)} = \frac{K}{\tau s+1} e^{-t_d s} \tag{6}$$

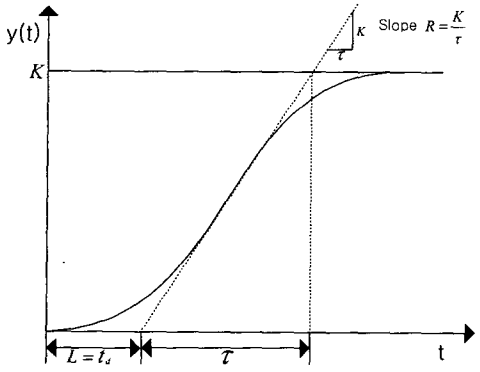

그림 7 지연이 있는 시스템 반응 곡선
Fig. 7 Reaction curve of the system with delay

표 1 Ziegler-Nichols의 PID 게인 튜닝
Table 1 Ziegler-Nichols PID tuning method

PID gain	K	T_I	T_D
Ziegler-Nichols optimum gain	$K=1.2/RL$	$2L$	$0.5L$

따라서, 본문에서 언급한 메시지 피드백 제어에서의 지연시간을 시간 지연의 매개변수로 고려해 볼 때, $t_d = t_{fcl}$으로 정의할 수 있으며, 시간지연 특성에 따라 PID 계수값을 결정하여 제어기의 응답특성을 보상할 수 있다.

4.2 모의실험 및 결과 분석

제어 대상시스템의 각 축은 선형운동을 하며 X, Y, Z로 표현되고, 속도정보를 제공하는 타코메타를 갖는 DC 모터로 인해 구동되어진다. DC 모터는 0~255사이의 제어 입력을 갖는 PWM 드라이브로 구동된다. PWM 입력과 위치정보의 출력간의 각 축의 전달함수는 다음과 같이 2차 선형시스템으로 표현된다.

$$G(s) = \frac{K}{s(\tau s+1)} \tag{7}$$

각 축의 시정수 τ(sec)는 0.055(X), 0.056(Y), 그리고 0.040(Z)이고 게인(gain) K((mm/sec)/PWM)는 각각 28.346 (X), 28.956(Y), 그리고 41.606(Z)이다[16].

이러한 시스템을 대상으로 피드백 제어 응답 특성을 살펴보기 위해서는 우선 각각의 노드에 따른 우선순위를 할당해야 하므로 사전 스케줄링이 선행되어야 한다. 따라서, 주 컨트롤러가 가장 우선순위가 높고 그 다음으로는 센서인 타코메타와 엔코더 그리고 액츄에이터 순으로 X, Y, Z축의 순서로 할당한다. 우선순위 할당에 따라 지연시간에 의한 피드백 제어의 응답 특성은 다르게 나타난다. 그림 8은 우선순위를 달리한 4가지 경우에 대한 지연시간 분석 결과를 보여준다.

그림 8은 각 노드에서 발생하는 메시지의 샘플링 주기(sampling period)는 5ms, 메시지 크기는 8바이트, 전송속도는 500Kbps, 그리고 모든 메시지는 동일한 시점에서 출발한다는 가정하에서 모의실험한 결과이다. 이러한 실험 결과로 알수 있는 사실은 각 노드에서 발생하는 메시지의 지연시간은 우선순위에 따라 다르게 나타난다는 것이다. 예를 들어, case 3(o)의 첫번째 노드에서 발생하는 메시지의 우선순위는 3이고, 지연시간은 약 851 μs가 소요된다. 우선순위가 3인 메시지를 발생시키는 노드를 그림 8에서 살펴보면, case 1(*)의 3번째 노드, case 2(+)의 6번째 노드, 그리고 case 4(x)의 6번째 노드가 같은 양만큼의 지연시간이 소요된다는 것을 볼 수 있다. 따라서, 동일한 가정 조건이라면 메시지 지연시간은 우선순위에 따라 상수로 가정할 수 있다는 것을 알 수 있다.

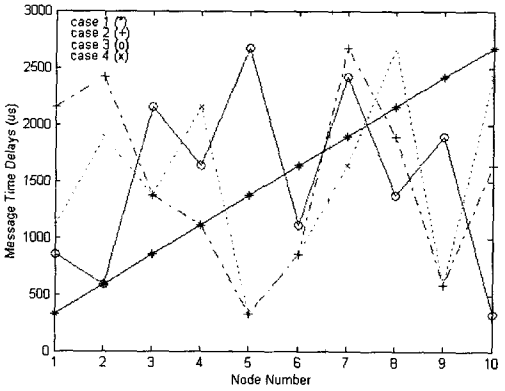

그림 8 4가지 경우의 우선순위 부여에 따른 메시지 지연시간
Fig. 8 Message delay time according to an identifier assignment for four cases

모터 제어기의 설계 사양을 최대오버슈트(M_p)는 28%이하, 2% 정착시간(T_s)은 0.5(sec)이하로 하고, 그리고 감쇠비

(ζ)를 0.5로 설정하고 근궤적법을 이용하여 PID 제어기를 설계하였을 때, 네트워크에 연결되어 있지 않은 상태에서의 각 축의 플랜트 스텝 응답을 측정한 결과는 그림 9와 같다.

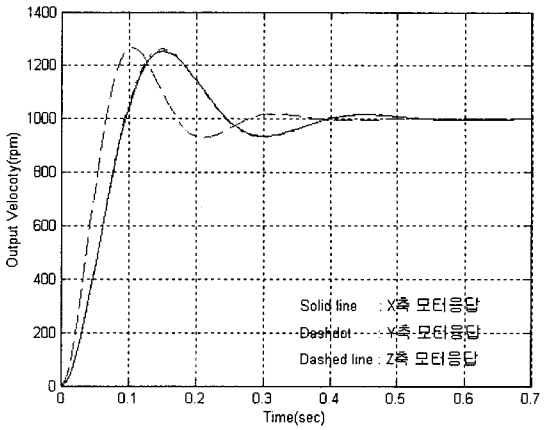

그림 9 각 축의 모터 응답특성
Fig. 9 Response characteristics of each axis' motor

그림 9에서 X축의 경우 M_p=26%, T_s=0.37(sec), Y축의 경우 M_p=26%, T_s=0.355(sec), 그리고 Z축의 경우 M_p=26%, T_s=0.36(sec)로 나타났으며 설계된 PID 제어기의 설계 사양을 만족하고 있다.

각 축의 시간지연 특성은 주 컨트롤러로부터 액츄에이터를 구동하기 위한 제어 메시지를 전달하는데 걸리는 지연시간이다. 그리고 본문에서 언급했듯이 각 노드에서의 메시지 우선순위에 따라 다른 지연시간을 갖는다.

그림 10, 11의 모의실험 결과들은 모의실험 전반부에 할당한 메시지 우선순위에 의해 나타난 결과이고 X,Y축의 응답특성이 비슷하여 X, Z축의 결과만을 분석해 보여주고 있다. 그림 9, 10에서의 응답특성 곡선은 네트워크에 연결되지 않은 상태에서의 각 축의 응답특성, 네트워크에 연결되었을 때 시간지연이 있을 때의 응답특성, 그리고 시간지연을 고려하여 PID 계수를 조정했을 때의 응답특성 결과들을 나타낸다.

지연시간을 고려하지 않고 제어기를 설계했을 때는 다시 말해서 네트워크 지연이 있는 응답특성을 분석해 보았을 때, 그림 10의 점선(dotted line)으로 나타난 것처럼 정착시간은 0.38(sec)로 설계사양을 만족하지만 최대 오버슈트가 31%로 설계 사양을 만족하지 못하는 결과를 초래했다. 그러나, PID 계수 조정을 통해 보상한 후의 응답특성 결과가 대쉬도트 선(dashdot line)으로 나타난 것 처럼 상승시간은 다소 지연 되었지만, 최대오버슈트가 26%이고 정착시간도 설계된 제어기의 설계 사양을 만족하는 결과를 보였다. 위의 결과에서, 네트워크에 연결되었을 때 나타난 응답특성과 시간지연이 없는 제어기의 응답특성을 볼 때 상승시간에 있어 거의 비슷한 기울기로 올라가는걸 볼 수 있는데, 이것은 지연시간이 미소한 차이를 보이기 때문이다.

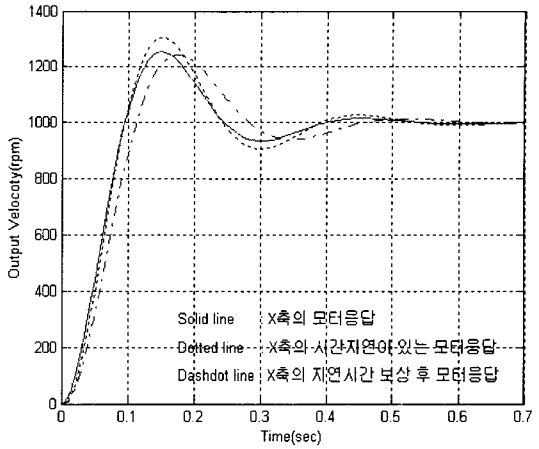

그림 10 X축의 시간지연에 따른 응답특성 비교
Fig. 10 Comparison of the X axis' response charateristics according to time delay

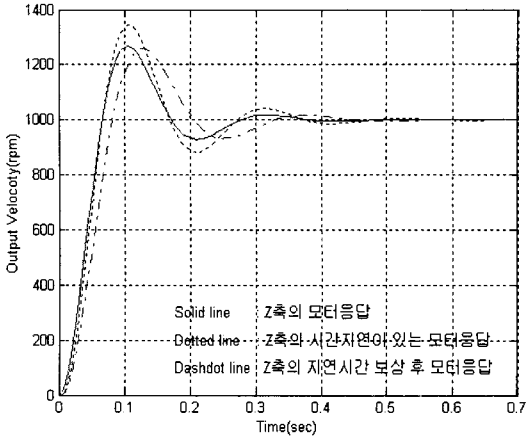

그림 11 Z축의 시간지연에 따른 응답특성 비교
Fig. 11 Comparison of the Z axis' response charateristics according to time delay

그림 11에서도 X축과 마찬가지로 네트워크 지연이 있는 응답특성의 결과가 점선으로 나타난 것처럼 정착시간은 0.42(sec)로 설계사양을 만족하지만 최대오버슈트가 34%로 설계 사양을 만족하지 못하는 결과를 보였다. 그러나, PID 계수 조정을 통해 제어기를 재 설계하여 응답특성을 보상한 후의 결과는 그림 10의 대쉬도트 선으로 나타난 것처럼 설계 사양을 만족시킬 수 있었다.

본 논문에서의 지연시간 분석 결과는 제어시스템을 설계시에 제어기의 시간지연 특성을 사전에 고려하여 제어기의 응답특성을 보상함으로써 시스템의 설계 사양을 만족시킬 수 있었으며, 시스템을 안정하게 설계할 수 있도록 하였다.

5. 결 론

네트워크를 기반으로 하는 제어시스템을 구성할 때, 메시지 전송에 따른 지연시간은 제어시스템의 불안정한 상태를 초래할 수 있다. 따라서, 네트워크 통신상에 발생할 수 있는 모든 지연시간 요소 분석이 선행되어야 하고, 지연시간을 고려한

제어기의 응답특성 분석을 통해 제어시스템을 설계해야 한다.

본 논문에서는 지연시간을 로컬 노드에서 발생하는 계산 지연시간과 메시지 전송을 위한 통신지연시간으로 정의하였고, CAN 통신 응용계층을 모델링하여 각각의 지연시간에 대한 구체적이고 정량적인 분석을 통해 메시지 지연시간을 상수로 고려할 수 있도록 수식적으로 재정의하였다.

유도된 수식과 분석된 로컬 영역에서의 메시지 지연시간을 제어기의 지연 특성으로 고려하여 응답특성을 보상함으로써 제어기의 응답특성을 개선시킬 수 있다는 것을 시뮬레이션 결과를 통해 알 수 있었다. 또한, 이러한 분석 결과를 통해 제어시스템 설계시 사전 스케줄링을 통해 각각의 노드에서 발생하는 메시지에 대한 지연시간의 정보를 미리 알 수 있어 시스템을 안정하게 설계할 수 있었다.

참 고 문 헌

[1] G. Schickhuber, and O. McCarthy, "Distributed fieldbus and control network systems," *Computing & Control Engineering Journal*, Vol. 8, No. 1, pp. 21-32, Feb 1997.

[2] R. Raji, "Smart networks for control," *IEEE Spectrum*, Vol. 31, No. 6, pp. 49-55, June 1994.

[3] Y. Halevi, and A. Ray, "Integrated Communication and Control Systems: Part I&II," *Transaction of ASME*, Vol. 110, pp. 367-381, Dec 1988.

[4] W. H. Kwon, Y. S. Kim., "Network Protocols in Distributed Real-Time System," *ICASE SPECIAL REPORT*, pp. 27-34, Nov 1998.

[5] M. Farsi, K. Ratcliff and M. Barbosa, "An overview of Controller Area Network," *COMPUTING & CONTROL ENGINEERING JOURNAL*, pp. 113-120, June 1999.

[6] G. F. Franklin, J. D. Powell, and A. Emani-Naeini, "Feedback Control of Dynamic Systems," Addison-Wesley, Reading, Massachusetts, third edition, 1994.

[7] 'CAN Specification 2.0', Parts A and B, Robert Bosch, Sep 1991.

[8] K. W. Tindell, A Burns, and A. J. Wellings, "Guaranteeing Hard Real-Time End-to-End Communications Deadlines," *RTRG/91/107*, Department of Computer Science, University of York, 1991.

[9] K. Tindell, A. Burns, "Analysis of Hard Real-Time Communications," *Technical report YCS 222*, Department of Computer Science, University of York, England, Jan 1994.

[10] K. Tindell, A. Burns, "Guaranteed Message Latencies for Distributed Safety-Critical Hard Real-Time Control Networks," *Technical report YCS 229*, Department of Computer Science, University of York, England, May 1994.

[11] K. Tindell, A. Burns, and A. Wellings, "Calculating Controller Area Network(CAN) Message Response Times," *Technical Report Y01 5DD*, Department of Computer Science, University of York, 1995. Available on line: http://cs.york.ac.uk.

[12] J. M. Jeon, and D. W. Kim, "An Analysis of Network-based Control System Using CAN(Controller Area Network) Protocol," *Proceedings of the 2001 IEEE International Conference on Robotics & Automation*, vol. iv, pp. 3577-3581, May 2001.

[13] J. K. Yook, D. M. Tilbury, and N. R. Soparkar, "A Design Methodology for Distributed Control Systems to Optimize Performance in the Presence of Time Delays", *Proceedings of the American Control Conference Chicago*, Illinoise, pp. 1959-1964, Jun 2000.

[14] CAN in Automation(CiA), "Implementation", http://www.can-cia.de/CANimpl.pdf.

[15] G. Cena, C. Demartini, and A. Valenzano, "On the Performanes of Two Popular Fieldbuses," *Factory Communication Systems, 1997. Proceedings. 1997 IEEE International Workshop*, pp. 177-186, 1997.

[16] L. Feng-Li, R. M. James, and D. T. Dawn, "Control Per formance Study of a Networked Machining Cell," *Proceedings of the American Control Conference Chicago*, Illinoise, vol. 4, pp. 2337-2341, June 2000.

저 자 소 개

전 종 만 (全 鐘 萬)

1973년 10월 15일생. 2000년 명지대학교 전기전자공학부 졸업. 2002년 동대학원 졸업(공석). 2002년~현재 (주)큐리텔 중앙연구소 연구원. 관심 분야는 실시간 시스템, 통신 네트워크, 통신 프로토콜, 인터넷 실시간 제어, 웹 컨텐츠.

Tel : (031)330-6472

E-mail : manytto@curitel.com

김 대 원 (金 大 元)

1960년 2월 15일생. 1984년 서울대학교 제어계측공학과 졸업. 1986년 동대학원 졸업(공석). 1990년 동대학원 졸업(공박). 1987년~1992년 대우중공업(주) 중앙연구소 선임연구원. 1992년~현재 명지대학교 정보공학과 교수. 관심분야: 자동화 네트워크, 실시간 시스템, 웹기반 응용

Tel : (031)330-6472

Fax : (031)330-6226

E-mail : dwkim@mju.ac.kr