'93 한국자동제어학술회의논문집 (1993.10.20~22)

이중 프로세서를 이용한 고장허용 제어기의 설계 및 실현

°최성규*, 홍일선**, 권오규*
*인하대학교 공과대학 전기공학과, **국방과학연구소

Design and Implementation of Fault Tolerant Controller Using Duplex Processors

°Seong-Kyu Choi*, Il-Sun Hong** and Oh-Kyu Kwon*
*Dept. of Electrical Eng., Inha Univ., **Agency for Defense Development

Abstract

In this paper, a fault-tolerant controller using duplex processors has been designed and implemented. The PI controller is adopted as the control alogorithm and the fault-tolerant control system is implemented by two single chip processors(MCS-96). Performances of the control system designed here have been shown via a simulation with application to a pitch channel autopilot.

1. 서 론

지난 20여년간에 저가의 고성능 마이크로프로세서와 컴퓨터의 개발및 보급에 따라 제어응용분야에 채택된 기법도 이를 활용하는 방향으로 크게 변모되어 왔으며, 디지탈 컴퓨터를 제어장치로 사용하는 이른 바 "컴퓨터 제어 시스템(Computer-Controlled System)" 이 독자적인 연구 분야가 될 정도로 성장하였다. 그런데 지금까지 컴퓨터 제어 시스템 분야에서는 주로 제어알고리즘의 실현상의 문제점들, 즉 이산화 오차, 양자화 오차, 다중 샘플링 문제 등의 처리에 연구가 집중되어 왔으며, 컴퓨터는 단순한 수치 연산장치로서의 기능만을 수행해왔다. 그러나 컴퓨터를 제어장치로 사용할 때에 단순한 수치연산장치로서만 사용하는 것은 컴퓨터의 성능을 충분히 활용하지 못하는 것이라 할 수 있다. 컴퓨터는 수치연산뿐만 아니라 논리판단및 감시기능도 갖고 있기 때문이다.

고장허용제어란 대상시스템에 고장이 발생하였을 때에도 안정도및 제어성능을 유지하도록 시스템을 구성하는 제어기법을 뜻한다. 고장허용 제어시스템은 대부분 디지탈 컴퓨터로써 실현되는데, 이 시스템에서는 컴퓨터의 수치연산뿐만 아니라 논리판단및 감시기능까지 활용함으로써 제어시스템의 신뢰도를 향상시킨다. 이 분야에 대한 학계의 연구는 최근에 시작되었으며, 현대제어및 계측기법들이 관련된 광범위한 연구 분야로서 비행체계의 고성능 유도조종장치라든지 원자로, 전력계통및 각종 대형및 고가의 공정이나 플랜트의 감시제어장치구성에 본격적으로 활용될 전망이다[1, 8, 9].

고장허용 제어시스템의 구성법으로는 하드웨어 여유도(Redundancy)를 늘리는 방법과 소프트웨어 여유도를 늘리는 방법등 크게 두가지가 있다. 이 중에서 하드웨어 여유도를 늘리는 방법은 제어기, 구동기, 감지기들을 각각 여러개씩 병렬로 쓰면서, 어떤 제어기와 구동기가 정상적인 동작을 하는지, 어떤 감지기의 신호가 믿을 만한지를 다수결에 의해 판단하여 고장허용제어를 수행하는 방법이다. 그리고 소프트웨어 여유도를 늘리는 방법은 제어기, 구동기, 감지기들의 특성이나 해석적 모델을 이용하여 각 부분의 고장 여부를 판단하면서 이에 상응하는 조치를 함으로써 고장허용제어를 수행하는 기법이다. 그런데 하드웨어 여유도를 늘리는 방법은 개념적으로는 매우 간단한 방법이지만 여유도가 커질수록 시스템이 복잡해지고 비용부담이 너무 커지기 때문에 이 분야에서의 최근의 연구는 가능한한 하드웨어 여유도는 작게 하면서 소프트웨어 여유도를 늘림으로써 신뢰도를 향상시키는 방향으로 진행되고 있는 추세이다[1].

이 논문에서는 단일 칩 프로세서 두개로 구성되는 고장허용 제어기를 설계및 구현하고자 한다. 여기에 쓰이는 두개의 프로세서는 주종관계의 프로세서(master-slave processors)로서 각각 고장 진단과 제어를 맡고, 또 고장이 나면 역할을 바꾸는 식으로 고장허용제어를 수행한다. 단일 칩 프로세서(one chip processor)로서는 MCS-96을 채택하고 제어대상시스템은 M-3H-3 발사체의 피치 채널(pitch channel)로 하고, 제어기로는 편의상 PI 제어기를 설계하고 MCS-96에 의해서 실현한다. 또한 가상의 고장주입(fault injection) 모의실험을 통하여 이 논문에서 구현된 제어기의 성능을 실험한다.

이 논문의 구성은 다음과 같다 : 제 2장에서는 M-3H-3발사체에 대한 이산화된 PI 제어기를 설계하고 제 3장에서는 이것에 대한 고장허용 제어기를 이중 프로세서에 의해 실현한다. 제 4장에서는 실현한 제어기의 단위계단 응답실험과 고장이 발생했을 때의 제어모습을 보이며 제 5장에서는 결론을 도출해낸다.

2. 제어 알고리즘

PID 제어기는 산업 현장에서 각종 공정 제어에 널리 활용되고 있는 제어기법으로서 일반적인 구조는 그림1과 같다.

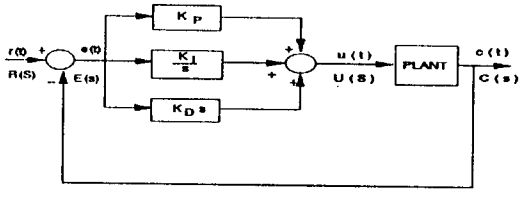

그림1. 일반적인 아날로그 PID제어기 구조

이 논문에서는 편의상 $K_D = 0$으로 놓고 PI 제어기를 제어 기법으로 채택한다. 기본적인 PI 제어기의 구조는 연속형 시스템에서 식(2 - 1)처럼 표시된다.

$$u(t) = K_P\, e(t) + K_I \int e(t)\, dt \qquad (2-1)$$

여기에서 u 는 제어 입력 변수이고 e 는 오차 신호이며 제어 변수는 오차 신호에 비례하는 K_P 항과 오차 신호의 적분에 비례하는 K_I 항으로 이루어진다.

식(2 - 1)의 연속형 PI제어기 구조를 속도형의 이산화된 형태로 나타내면 다음과 같다[3]:

$$u(k) = u(k-1) + (K_P + TK_I)e(k) - K_P\, e(k-1) \qquad (2-2)$$

이 논문에서는 제어기로서 식(2 - 2)의 제어 알고리즘이 사용된다.

PI 제어기의 하드웨어 부분은 단일 칩 프로세서, DA 컨버터, 메모리, 아날로그 회로로 이루어진다. i8097은 제어를 위한 16비트 단일 칩 프로세서로 10 비트 AD 컨버터를 내장 하고 있고 PC 와의 통신을 가능하게 하는 시리얼 포트를 내장하고 있다[7]. 또 DA 변환을 위해 8비트 전압 출력 특성을 가진 AD7224가 사용되고 전압 스케일링을 위한 아날로그 회로 등이 사용되며 프로그램과 데이터 보존을 위한 롬 과 램이 사용된다.

PI 제어기의 소프트웨어 부분은 식(2-2)를 Q8이라는 고정 계수 소수점 방식으로 표현하여 연산을 수행하도록 어셈블리 언어로 이루어진다.

3. 고장허용 제어기의 설계

3.1 고장허용 제어

일반적으로 제어시스템은 성능, 가격및 신뢰도를 고려하여 최적화하도록 설계하여야 한다. 고장허용 제어시스템에서는 하드웨어 여유도를 높이면 신뢰도는 높아지지만 가격이 상승한다는 단점이 있다. 따라서 이 논문에서는 원하는 신뢰도를 가지면서 낮은 가격과 성능의 희생을 최소화하기 위해 주종관계의 프로세서에 의해 고장허용 제어기를 실현한다[5].

주종관계의 프로세서를 사용한 고장허용 제어기의 개념은 다음과 같다 : 주프로세서(master processoor)와 종프로세서(slave processor)라는 두 개의 프로세서가 사용되는데, 먼저 종프로세서가 제어의 역할을 맡아 실제의 제어기의 출력(제어 입력, control input)을 낸다. 반면에 주프로세서는 종프로세서의 고장유무를 계속 지켜본다. 만일 종프로세서에 고장이 발견되면 종프로세서가 하던 제어를 중단시키고 주프로세서가 종프로세서가 하던 제어의 역할을 맡는다[6].

고장허용 기법은 고장의 검출(fault detection)과 자연스러운 차단(graceful degradation)과 재배치(reconfiguration)의 3단계로 구성된다[2]. 여기에서 고장 검출은 소프트웨어에 의한 각각의 하드웨어를 검사(built in test software)함으로서 이루어지는데 다음 절에서 그에 대한 기법들이 설명된다.

자연스러운 차단(graceful degradation)이라는 것은 하드웨어의 고장이나 소프트웨어 에러 부분을 자연스럽게 분리시켜 나가는 능력이다. 만일 자체 평가 소프트웨어에 의해 종프로세서에 고장이 검출되면 종프로세서가 하던 제어를 중단시키는 과정이 자연스러운 차단 과정(graceful degradation)이다.

재배치(reconfigration) 과정은 종프로세서를 차단시키고 주프로세서를 온(on) 시키는 것으로서 이 역할은 주프로세서의 명령에 의한 아날로그 멀티플렉서의 채널 변경에 의해 이루어진다. 제어대상시스템에 대해 종프로세서가 PI 제어를 행하며

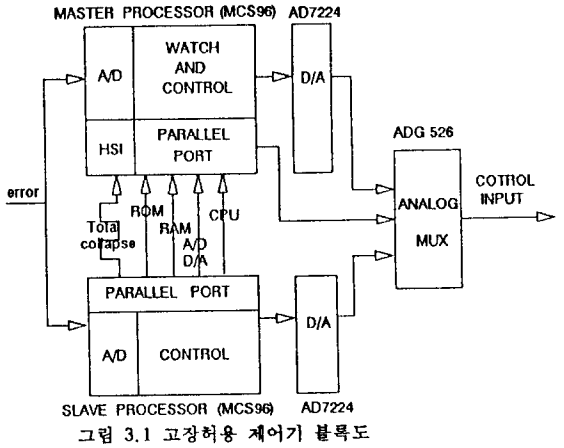

그림 3.1 고장허용 제어기 블록도

자체 평가 소프트웨어에 의해 고장을 감시하고 고장유무를 병렬포트로 전달한다. 주프로세서는 병렬포트와 고속입력부(HSI)에 의해 종프로세서의 병렬포트(parallel port)를 감시한다. 이때 종프로세서의 고장이 주프로세서에 의해 인식되면 주프로세서는 아날로그 멀티플렉서를 종프로세서에서 차단(off) 시키고 주프로세서로 변경한다.

그림 3.1이 이 논문에서 채택한 이중프로세서에 의한 제어기의 블록도 이다.

3.2 제어기 고장 점검

제어기의 고장은 시스템이 전혀 동작하지 않는 전체 고장과 일부분의 하드웨어 고장에 의하여 오동작을 하는 부분 고장으로 나누어서 고장을 점검 할 수 있다.

우선 부분적인 고장은 자체평가 소프트웨어에 의해 검출된다. 자체 평가 소프트웨어란 주어진 하드웨어의 고장을 진단할 목적으로 사용한 소프트웨어이다. 자체평가 소프트웨어는 MCS-96 같은 단일 칩 프로세서에서는 특별히 중요하다. 왜냐하면 A/D 컨버터의 멀티플렉서의 한 채널이 고장이 났다면 소프트웨어에 의해 한 채널을 차단(off) 시키고 다른 채널로 변경할 수 있기 때문이다.

다음은 하드웨어 다양한 구성 요소에 대한 자체평가 소프트웨어를 구성요소별로 나누어 설명하기로 한다.

가. CPU 점검

CPU의 정상 동작은 다른 어떤 하드웨어 구성요소 보다 중요하다. 여기에서는 MCS-96의 수치 논리 연산부(ALU)와 플래(FLAG)을 체크함으로서 CPU의 고장 유무를 판단한다. 다음과 같은 방법에 의해 판단된다. 먼저 5555H - 5555H 라는 뺄셈 연산을 수행한 다음 제로 플랙이 0인지 아닌지를 검사한다. MCS-96의 논리상 0로 클리어(clear)되어야 정상이므로 만약 0이 아니면 CPU의 고장으로 볼 수 있다. CPU가 고장임이 확인되면 종프로세서는 병렬포트(P1.4)를 1로 세트(set)시켜 주프로세서에게 알린다. 주프로세서는 병렬포트로 종프로세서가 고장인지 아닌지를 지켜 보다가 고장이면 아날로그 멀티플렉서의 채널을 변경시켜 주프로세서가 PI 제어를 맡게 한다. 이런 방식으로 덧 뺄셈을 반복함으로써 오버플로우 플랙, 캐리 플랙 등을 점검하고 CPU의 이상유무를 확인할 수 있다.

나. 메모리 점검

비휘발성 메모리를 포함한 모든 메모리는 전기적 방전이나 갑작스런 전원의 변동, 진동, 또는 그 밖의 다른 수단에 의해 세포가 파괴되어 고장을 일으킬 수 있는데 메모리의 한 두세포의 파괴는 바로 제어기의 완전한 고장으로 이끌지 않지만 반복 연산을 통해서 끝내는 제어를 못하게 하는 원인을 만든다. 그러므로 여기에서 사용한 롬(ROM)과 램(RAM)의 고장에 대한 대책이 필요하다.

롬의 고장 비트는 체크 썸(check sum) 이라는 방식으로 알고 있는 체크 썸과 현재 계산한 체크 썸을 비교하여 검출한다. 일반적으로 모든 롬 영역의 프로그램 코드의 이진합인 체크 썸은 프로그램을 링크시킬때 계산하여 롬의 어떤 특정한 영역에 보관하여 이 보관된 값과 프로그램 코드와의 차가 0인가를 확인하여 고장 유무를 판단한다. 만약 고장이 발생하면 CPU 고장 때와 똑같은 방식으로 주프로세서가 제어를 담당한다.

롬과는 달리 램은 쓰고 지울 수 있는 특성 때문에 체크 썸 방식이 이용 될 수 없다. 때문에 쓰고 지우는 특성을 이용한 재순환 점검방법을 이용한다. 먼저 램 영역을 반으로 나누고 위 쪽 램 영역에 모두 FFH로 쓴 다음 다시 읽는다. 읽은 데이터가 모두 FFH이면 다시 위 쪽 램 영역에 0H를 쓴다. 다시 읽어서 모두 0H이면 이상이 없는 것으로 여긴다. 이상이 없으면 PI 계수나 임시 저장 변수, 계산 값 등등의 데이터 값은 위 쪽 메모리에 저장한다. 여기에 다시 FFH, 0H 를 쓴다면 데이터를 모두 상실하므로 대피시킬 장소가 필요하다. 지금 아래쪽 반을 사용하지 않았으므로 아래쪽 반을 FFH, 0H을 쓰고 읽어서 이상이 없음을 확인하고 데이터 값을 대피시키고 위쪽 메모리의 고장유무를 확인한다. 만약 고장난 세포가 있다면 CPU나 롬이 했던 방식대로 주프로세서에게 고장을 알린다.

다. A/D, D/A 점검

제어입력 데이터는 D/A 컨버터를 통하여 제어대상시스템의 입력으로 들어가므로 이 D/A 컨버터의 값을 다시 A/D 컨버터로 읽어 그 값들을 비교해 봄으로서 고장유무를 알아 낼 수 있다. MCS-96에는 A/D의 채널이 8개나 되므로 제어기의 입력인 에러 신호를 받아들이는 A/D의 한 채널을 제외 한 7개중의 하나를 사용하여 값을 읽어 비교하면 된다. D/A는 8비트이고 A/D는 10비트이므로 모두 8비트로 환산하여 아날로그 잡음 등을 고려해 8비트의 범위 255중에서 비교한 값에서 10정도의 차이는 고장이 없는 것으로 한다. 만약 A/D, D/A를 비교한 값이 10 이상이면 고장인 것으로 간주하여 앞에서와 같은 방식으로 주프로세서에게 알린다.

라. 와치 도그 타이머(watchdog timer)

와치 도그 타이머는 소프트웨어의 잘못에 의한 자연스러운 차단(graceful degradation)의 수단으로 사용되며 MCS-96에서 제공된다. 소프트웨어로 주기적으로 와치 도그 타이머 레지스터를 클리어시키게 한다. 만약 프로그래밍이 잘못되거나 외부의 간섭 등에 의해 프로그램 카운터가 갑작스럽게 변동을 일으켜 클리어 시키는 부분을 돌지 않으면 타이머가 오우버플로우를 일으켜 CPU를 강제로 재동작(reset) 시킨다.

다음은 시스템이 완전히 고장나서 전혀 동작하지 않을 때의 검사 방법이다. 종프로세서에서 샘플링 시간마다 주기적인 펄스를 보내도록 프로그램 한다. 주프로세서에서는 고속 입력 포트에서 정해진 샘플링 시간에 펄스가 들어오면 가로채기(interrupt)가 걸리게 해서 계속 종프로세서의 고장유무를 판단

하게 하고 펄스가 들어오지 않아 가로채기(interrupt)가 걸리지 않으면 종프로세서를 차단(off) 시키고 주 프로세서가 제어의 임무를 하게끔 한다.

4. 모의실험및 결과

이 논문에서 구현된 고장허용 제어기의 성능을 시험하기 위해 모의실험을 수행한다. 제어대상시스템으로는 다음과 같이 표시되는 M-3H-3발사체의 피치 채널을 선택하였다.

$$G_P(s) = \frac{5.985s + 0.000286}{s^2 + 0.01825}$$

이 피치 채널의 선형화한 전달함수를 PC에서 미분 방정식을 풀어 필요한 입력과 출력을 A/D와 D/A를 통하여 처리함으로써 이루어진다[4].

실험은 모의 발사체의 피치 채널에 대한 종프로세서에 의한 단위계단 응답실험과 고장주입(fault injection) 실험에 의한 종프로세서에서 주프로세서로 제어가 넘어가는 과정을 보인다.

먼저 그림 4.1은 피치 채널의 개루프 단위계단 응답특성을 나타내고 그림4.2는 폐루프 단위계단 응답특성을 나타낸다. 그림4.3은 프로세서를 사용하지 않고 PI 계수변화에 의한 단위계단 응답의 모의실험을 보인 것이다. 폐루프 응답과는 달리 만족할 만한 성능을 보이고 있다. 그림4.4는 종프로세서에 PI 제어 루틴을 실어서 모의 발사체의 피치 채널에 자세 각 1 도를 준 응답특성을 나타낸다. 빠른 응답과 오버슈트 0, 정상상태 오차 2% 이내의 아주 만족할 만한 성능을 보이고 있다. 이때의 PI 계수는 각각 3.333, 13.7 이다. 다음 그림 4.5는 PI 계수 변화에 따른 응답특성을 나타낸다. 그림에서 보는 것처럼 I 계수가 크면 클수록 정착 시간이 빨라짐을 알 수 있다. 하지만 I 계수가 어느 한도 이상 커지면 발산해 버림을 실험으로 알 수 있었다.

그림 4.6 은 고장주입(fault injection) 실험이다. 제어기에 3.7초에 임의로 고장을 일으켜 성능변화를 살펴보았다. 3.7초까지는 잘 돌아가던 시스템이 3.7초 이후에는 급격히 발산해 버림을 알수있다(주프로세서를 사용하지 않는 경우임). 따라서 이러한 고장에 의해 발생하는 불의의 사태를 예방하기 위하여 고장허용 제어기가 설계 된 것이다. 그림 4.7 은 주종프로세서를 사용한 제어기에 고장주입 실험을 한 것이다. 종프로세서가 제어하다가 고장때문에 발산하려한다. 이때 수동적인 조작에 의해 종프로세서를 차단 시키고 주 프로세서가 제어를 담당하게 한다. 제어를 시작한지 몇 상태의 시간이 지나면 다시 정상 궤도에 오르게 된다. 하지만 이 실험은 고장이 나면 사람이 직접 모드 변경을 해야 한다는 불편함이 생기고 이에 따른 위험 부담이 있다. 그래서 모드 변경의 역할을 주 프로세서가 하도록 설계하여 고장 주입 실험을 한 결과 그림 4.8과 같은 좋은 제어 성능을 발휘함을 알 수 있었다. 약간의 변화가 있는 부분이 고장주입 실험을 한 부분이다.

5. 결론

이 논문에서는 단일 칩 프로세서를 이중으로 써서 고장허용 제어기를 설계하고 실현하였으며, 이것을 발사체의 피치 채널 제어에 적용하고 모의실험을 수행하여 그 성능이 만족할 만함을 확인하였다. 이 논문에서 채택한 제어 알고리즘은 PI제어기이나 소프트웨어로써 다른 제어 알고리즘도 쉽게 구현시킬 수 있다. 추후과제는 제어기 뿐만 아니라 제어대상체의 고장을 진단하여 제어대상체가 고장을 내포하더라도 그에 알맞게 제어를 할 수 있는 총체적인 고장허용 제어기를 설계하고 실현시키는 것이다.

참고 문헌

[1] K. Warwick and M. T. Tham, Failsafe control systems, Chapman and hall, pp.1-4, 1989.

[2] B. W. Johnson, Design and analysis of fault tolerant digital systems, Addison wesley publishing company, pp.1-2, pp.52-54, pp.24-30, 1989.

[3] G. A. Perdikaris, Computer controlled systems theory and applications, pp.395-397, 1991.

[4] 이 명의, A study on design and realization of fault tolerant digital autopilot, 1991.

[5] J. J. Serrano, C. Cebrian, J. Vila and R. Ors, A low cost fault tolerant multiprocessor system for real time control, IFAC Low cost Automation pp.441-445, 1986.

[6] P. Laplante, Real time systems design and analysis, IEEE PRESS pp.253-270, 1992.

[7] Intel, Embedded controller handbook, Intel coperation, pp. 12.1-16.103, 1987.

[8] R. F. Stengel, Intelligent failure-tolerant control, IEEE Control Systems Magazine, vol.11, No.4, 1991.

[9] R. J. Patton, P. M. Frank and R. N. Clark ed. , Fault Diagonis in Dynamic Systems, Prentice-Hall Inc. , U. K. , 1989.

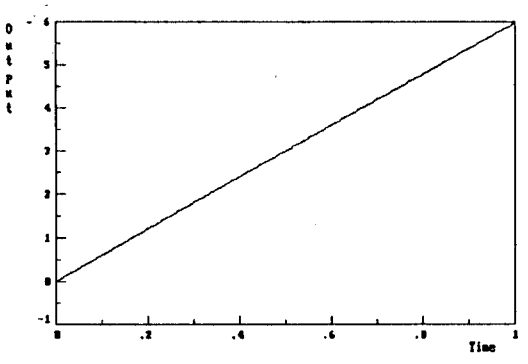

그림 4.1 개루프 단위계단 응답

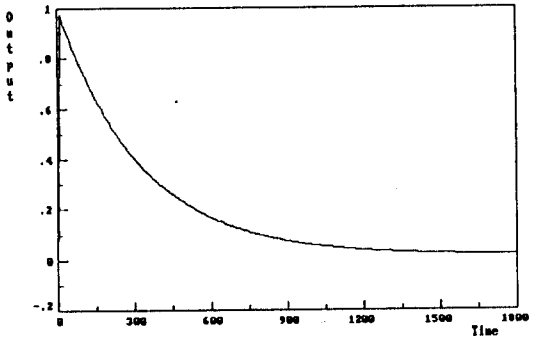

그림 4.2 폐루프 단위계단 응답

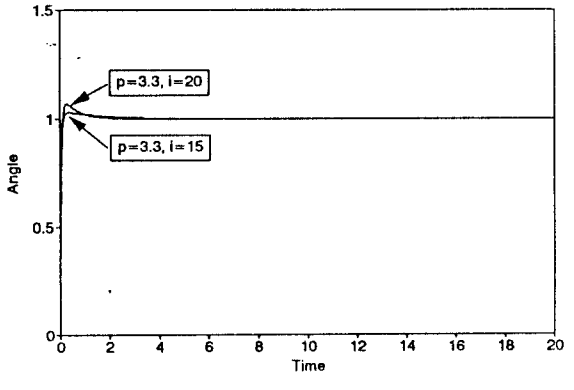

그림 4.3 PI제어 모의실험

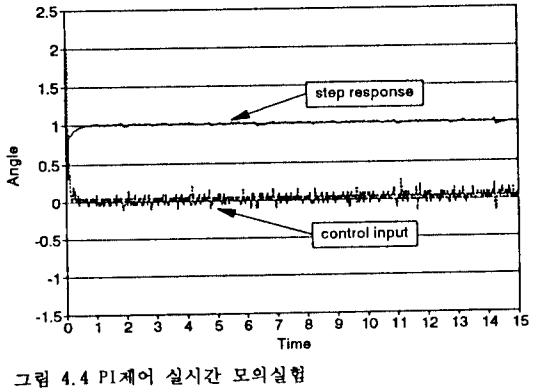

그림 4.4 PI제어 실시간 모의실험

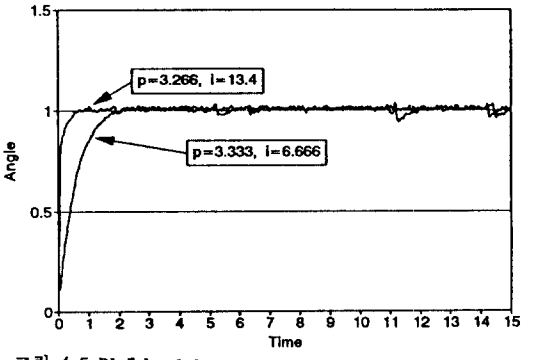

그림 4.5 PI계수 변화에의한 단위계단 응답 특성 변화

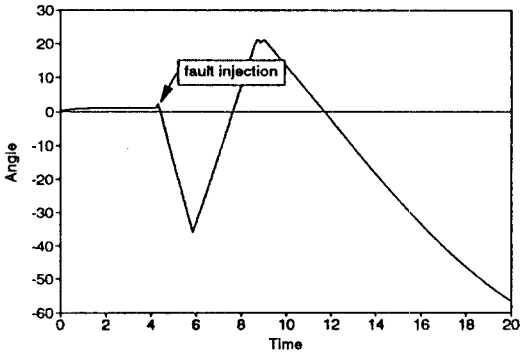

그림 4.6 고장주입 실험

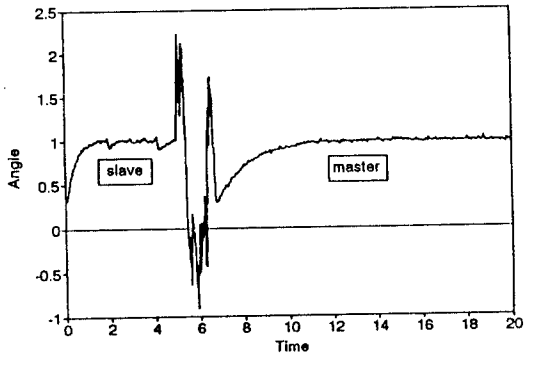

그림 4.7 수동조작에의한 고장허용실험

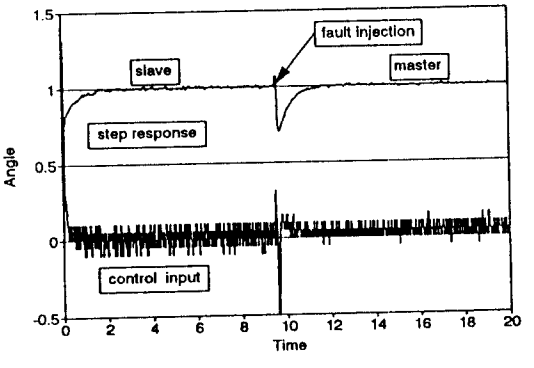

그림 4.8 자동변환에의한 고장허용실험