論文95-32B-12-1

제철소 연속주조 공정에서의 퍼지제어를 위한 기능코드의 구현 연구

(A Study on Realization of Function Code for Fuzzy Control in the Continuous Casting Process of the Iron & Steel Works)

許 倫 紀 * , 朴 世 華 **, 李 在 爀 ***, 卞 增 男 ****

(Y.-G. Hur, S.-H. Park, J.-H. Lee, and Z. Bien)

요 약

현대 산업 공정이 점점 더 복잡해짐에 따라, 플랜트의 모델을 구하고 그의 공정을 제어하기가 더욱 어렵게 되어가고 있다. 따라서, 현재 사용되고 있는 분산 제어 시스템(DCS)을 보다 고신뢰화하고 고기능화한 진보된 형태의 분산 제어 시스템으로 전환해야 할 필요성이 대두되었다. 고급 분산 제어 시스템(Advanced DCS)이란 기존의 분산 제어 시스템에 보다 진보된 기능을 첨가한 DCS를 말하며 선진국에서도 최근에야 개발되고 있는 분야이다. 예를 들어, 고장진단 및 고장대처 기능 그리고 GPC(Generalized Predictive Control), NN(Neural Network), 퍼지제어 기능이 첨가된 시스템을 말한다. 본 논문에서는 지능제어의 한 형태인 퍼지제어 알고리즘을 DCS에 구현하는 방법에 관하여 연구하였다. 퍼지제어 알고리즘은 산업체 기술자들이 사용하는 공정 제어 언어인 기능코드의 형태로 구현되었고, 구현된 퍼지제어 기능코드의 검증을 위하여 포항제철 광양제철소의 연속주조 공정을 대상으로 하여 퍼지제어기를 구성하여 퍼지제어의 효용성을 확인하였다. 본 퍼지제어에 사용된 규칙은 현장 조업자들과의 면담과 제어에 관련된 자료들로부터 얻었으며, 퍼지제어 기능코드의 유용성을 실시간 운영체계의 환경에서 모의 실험으로 보인다.

Abstract

As the modern industrial processes become more complex, it is getting more difficult to model and control the processes. Naturally, an advanced type of DCS(Distributed Control System) with higher level functions is being sought. Advanced DCS is a DCS with advanced functions such as fault diagnosis, GPC(Generalized Predictive Control), NN(Neural Network), and Fuzzy Control. In this thesis, we have studied a fuzzy control algorithm for realizing an advanced DCS. Its algorithm is implemented in a form of function code which is a process control language, being used by the industrial engineers. To verify the realized function code of the fuzzy control, the function code is applied to a continuous casting process of the Pohang Iron & Steel Works in Kwangyang. The rules of the fuzzy control were collected via interviews of the field operators and their operation documents. Finally, under a real-time operating system environment, usability of the function code of the fuzzy control is shown via simulation for the continuous casting process.

* 正會員, 浦港製鐵(株) 技術研究所
(POSCO Tech. Research Lab.)
** 正會員, 生産 技術 研究員
(Korea Academy of Industrial Technology)
*** 正會員, 韓國 外國語大學校 制御計測工學科
(Dept. of Control & Instrumentation, Hankuk Unv. of Foreign Studies)
**** 正會員, 韓國科學技術院 電氣 및 電子工學科
(Dept. of Elec. Eng., KAIST)

接受日字: 1994年2月23日, 수정완료일: 1995年12月4日

Ⅰ. 서 론

발전소와 제철소 등과 같은 대규모 공정 제어 분야에 디지탈 계장제어 시스템이 도입되고 있으며, 과거 아날로그 제어방식에 비해 분산 제어 시스템(DCS, Distributed Control System)은 제어 시스템의 안정성 있고 효율적인 운용을 위해서 필요하며 그 중요성은 강조하지 않아도 이미 산업체에서 많이 인식되어 있다[2]. DCS는 (그림 1)에 개념을 간략화 하여 도시하였으며 간단히 설명하면 다음과 같다. DCS는 OIS(Operator Interface Station)와 EWS(Engineering Work Station)과 PCS(Process Control System)의 3부분으로 구성되어 있다. OIS는 오퍼레이터와 공정이 운용되고 있는 하드웨어(PCS)간의 인터페이스를 담당한다. 오퍼레이터는 공정에서 운용되는 설정치(SV, Set Value) 및 제어입력(MV, Manipulated Value) 그리고 공정 출력값(PV, Process Value) 등을 모니터링한다. 또한 알람 신호 및 제어기의 파라메터 값을 설정하기도 한다. EWS는 엔지니어가 제어 알고리즘을 구현하는 곳이다. 본 연구에서 구현하고자 하는 제어 알고리즘은 이 곳 EWS에서 이루어진다.

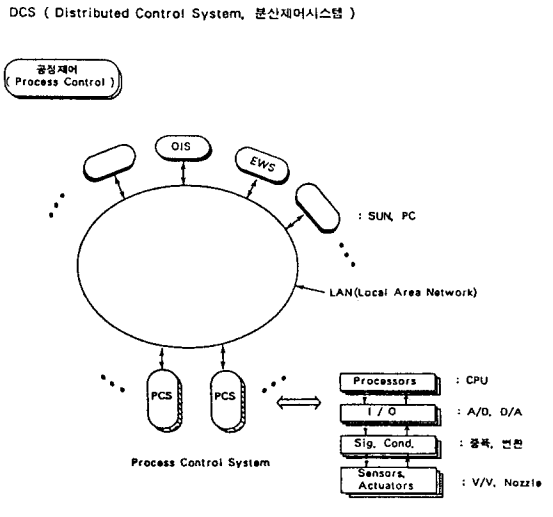

그림 1. 분산 제어 방식
Fig. 1. DCS(Distributed Control System).

본 연구에서는 DCS중에서 퍼지제어 알고리즘을 구현하고자 한다. 제어 알고리즘을 DCS에 구현하려면 현재 실무자들이 사용하고 있는 공정 제어 언어인 기능코드의 형태가 되어야 하는데, 이는 각 기능별로 서브루틴(subroutine)을 작성해 놓으면 사용자는 자신이 구현하고자하는 제어 알고리즘에 맞게 그의 서브루틴인 기능코드를 이용하여 사용하는 방식이다. 기능코드화 해야하는 이유는 사용자가 편리하게 제어 알고리즘을 구현하고 제어 알고리즘 구현의 일반성을 부여하기 위해서이다. 기능코드의 나열로써 컨피그레이션 파일(configuration file)이 구성되고 이 파일에는 제어 알고리즘이 담겨있다. 여기서 기능코드를 작성한다는 말뜻은 컨피그레이션 파일의 각각의 함수(function)에 해당하는 서브루틴(subroutine)을 코딩한다는 말이다. 여기서 코딩은 컴퓨터언어 중에서 프로세서에 이식성이 용이한 C언어로 하였다[2].

기존의 기능코드에 비하여 신경망과 퍼지같은 지능제어를 기능코드화 하려면 기존의 기능코드에 기능이 부가되어야 한다. 기존의 기능코드는 스칼라(scalar) 값을 입력받아서 스칼라값을 출력하는 데 반하여, 지능제어의 기능코드는 벡터와 행렬을 다룰 수 있어야 하고 또한 그의 크기(size)도 사용자가 자유로이 선정할 수 있어야 한다.

구현된 퍼지제어 기능코드의 검증을 위하여 제철소 연주 공정(Continuous casting process)을 대상으로 하였다. 연주 공정은 용해된 강(steel)을 가지고 압연 공정에서 가공하기 적합한 주편(slab)을 만드는 공정이다. 이 공정을 거침으로써 쇳물이 형태를 가진다. 현장의 경우 정상상태는 PID 자동제어로 운용되고 있으나, 주조 시작과 종료는 숙련조업자의 수동제어로 이루어지고 있다. 따라서 공정의 완전 자동화를 위해서 주조 시작과 종료를 자동화할 필요가 있다. 주조 시작과 종료의 자동화는 성력화 및 조업안정화와 품질향상 그리고 생산성향상을 위해서 중요한 작업이다. 본 연구에서는 연주 공정의 주조 시작을 퍼지제어로 모의실험한다. 구현된 퍼지제어에서의 제어 규칙은 현장 프로세서 컴퓨터의 데이타와 조업자와의 면담(interview)을 통하여 수집하였다. 기능코드화된 퍼지제어 알고리즘으로 대상시스템을 실시간으로 실험한다.

본 논문의 구성은 다음과 같다. I장의 서론에 이어서, II장에서는 DCS에 퍼지제어 알고리즘을 구현하는 방법 및 기능코드에 대해서 서술한다. III장에서는 퍼지제어 기능코드의 검증을 위하여 제어대상인 제철소 연주 공정에 대하여 살펴본다. IV장에서는 연주 공정의 퍼지제어기 구성과 정상상태 PID 제어기와 전환방법을 설명하고 제어규칙을 소개한다. V장에서는 연주

공정을 대상으로 퍼지제어를 기능코드로 실험한 것을 소개한다. VI장에서는 본 논문의 연구결과를 정리하였다.

Ⅱ. DCS에 대한 퍼지제어 알고리즘 구현방법

1. 기능코드의 개요

기능코드란 공정 제어 언어이며, 제어기의 구성은 기능코드의 나열로서 이루어진다. 이때 기능코드의 나열로서 이루어진 파일을 컨피그레이션 파일이라 하며, 이러한 파일은 컴퓨터 시스템이나 타겟(target) 프로세서(processor)가 알 수 있는 코드로 바뀌어야 한다. 이 경우 프로세서와의 이식성을 고려해 볼 때 타겟 프로세서에 독립적인 코드인 C언어가 대표적이며 널리 사용되고 있다[2]. 엔지니어가 제어 알고리즘을 손쉽게 구현하기 위하여 그래픽 에디터(graphic editor)에 기능블록을 그려서 제어 다이어그램을 구성한다. 구성이 완료된 후, 이를 저장하면 (그림 2)의 예와 같은 컨피그레이션 파일이 생성된다. 이를 트랜스레이터(translator)라 불리는 제어언어 생성기를 거치면, C 파일이 생성된다. 여기서 트랜스레이트는 lex과 yacc이라 불리는 두 단계를 거친다. 생성된 C 파일을 컴파일(compile)하여 PCS에 다운로드(download)하면 엔지니어가 생각한 제어 알고리즘이 실제 공정에서 운용된다.

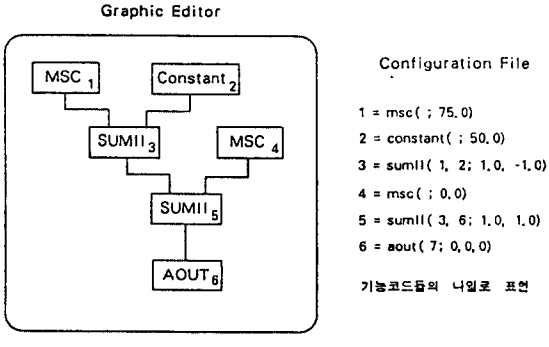

그림 2. 그래픽 에디터 및 컨피그레이션 파일
Fig. 2. Graphic editor and Configuration file.

2. 기능코드의 구현방법

공정의 규모가 커지면 제어 알고리즘이 그만큼 복잡하게 된다. 따라서 기능코드의 형태로 모듈화되어 있지 않으면, 새로운 알고리즘을 첨가하기 어렵고 또한 기존의 알고리즘을 수정하기도 쉽지 않다. 기능코드로 구현되기 위해서는 범용이 되어야 하고 따라서 기능코드간의 코딩 규칙이 있다[3]. 지능제어 알고리즘을 위한 기능코드의 특징은 기존의 기능코드는 스칼라(scalar) 값을 입력받아서 스칼라 값을 출력하는데 반하여, 벡터나 행렬을 수행할 수 있는 기능이 주어져야 한다. 예를 들어, 신경망같은 경우 가중치(weight)를 다루므로 벡터의 기능이 주어져야 하고, 퍼지제어는 퍼지한 값을 다루므로 벡터나 행렬을 처리할 수 있는 기능이 주어져야 한다. 또한 벡터나 행렬의 크기(size)가 변화가능해야 한다. 따라서 그의 크기가 유동적으로 변화가능해서 사용자가 마음대로 지정할 수 있어야한다. 다음은 기능코드의 표현형식에 대하여 소개한다.

블록번호 출력변수 = 기능코드이름 (입력 리스트 ; 파라메터값 리스트)

여기서 리스트는 해당 항목이 없거나 하나 혹은 여러 개가 될 수 있다. 항목이 없을 경우는 비워두고, 여러 개인 경우는 컴마(,)로 항목들을 구분한다. 각 부분에 대한 규칙 및 설명은 다음과 같다.

1) 블록번호 : 기능블록의 나열로써 제어기를 구성할 때, 각 블록에 할당되는 음이 아닌 정수이다. 이는 수행순서를 나타내고 내부 파라메터의 메모리 할당을 위해서 필요하다.
2) 출력변수 : 출력 포트를 나타내는 문자 변수이다.
3) 기능코드이름 : 기억하기 쉽고 기능을 적절히 표현할 수 있는 문자열로써, 각각의 기능을 갖는 기능코드에 고유한 이름을 갖는다.
4) 입력 리스트 : 기능코드의 입력을 나타낸다.
5) 파라메터값 리스트 : 기능코드를 수행되기 위하여 사용하는 파라메터 값들을 말한다.

위와 같은 규칙으로써 기능코드들을 구현해야 하며, 그로 말미암아 효과적인 제어 언어로서의 기능을 가지는 것이다.

퍼지제어 기능코드의 구현에 있어서 고려된 사항은 다음과 같다. 먼저 구조체(Structure) 구문을 사용할 수 없다. 이는 다른 기능코드에 영향을 주지 않으면서 기능코드 작성 규칙을 따른 것이다. 구조체를 사용할 수 없기에 규칙(Rule)과 소속함수의 형태가 행렬의 형태로 표현된다. 따라서 계산에 소요되는 시간이 많아진다. 이를 해결하기 위하여 퍼지제어 중에서 가장 간단

한 구조인 퍼지화는 이등변삼각형법을 사용하였고 추론은 MIN-MAX방법을 사용하였으며 사용하지 않는 규칙은 프로그램의 규칙 합성 과정에서 제거하여 추론에 소요되는 시간을 줄였다. 전체집합(Universe of Discourse)은 사용자가 마음대로 지정이 가능하다. 또한 규칙의 튜닝(Tuning)이 용이하도록 규칙이 자료파일(Data file)의 형태로 입력받도록 되어 있다.

Ⅲ. 연주 공정 시스템의 개요

제철소는 크게 제선 공정, 제강 공정, 그리고 압연 공정의 3단계를 거친다. 연주 공정은 제강 공정과 압연 공정을 연결해 주는 공정으로서, 제강 공정에서 나온 용강을 압연 공정에 적합한 주편(Slab)을 만드는 공정이다. 과거에는 쇳물에서 주편을 만들기까지 여러 과정을 거쳤으나 현재는 연속적으로 주조하고 있다. 현재 포항제철의 연주 공정의 조업자중 주조 속도 담당과 주형(Mold)내 용강의 레벨 제어하는 것은 고도의 숙련된 조업자가 하고 있다. 이는 공정의 안전성과 제품의 품질을 위해서 중요하다. (그림 3)은 포항제철 광양제철소 제 1연주 공장의 연주 공정을 나타낸다.

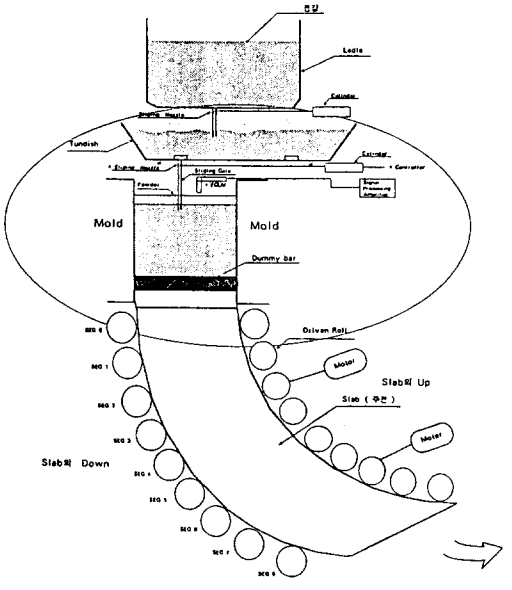

그림 3. 제철소 연속주조 공정
Fig. 3. Continuous Casting Process in the Iron & Steel Works.

연주 공정에서 제어의 중요한 부분은 주형내 용강의 레벨을 제어하는 것이다. 레벨제어가 제대로 이루어지지 않으면 제어시 큰 오버슈트(overshoot)는 주형 밖으로 쇳물의 분출(overflow)을 야기하여 사고의 위험을 초래한다. 또한 심한 댐핑(damping)은 주형내 용강의 요동을 야기하여 주편의 표면에 손상이 가서 제품의 품질을 저하시킨다. 따라서 제어가 잘되려면 안정되게 용강의 레벨이 원하는 레벨에 도달해야 한다.

본 연구의 과도상태의 레벨제어는 숙련조업자의 애매한 지식을 적절하게 이용가능한 퍼지제어로 수행한다.

Ⅳ. 연주 공정의 퍼지제어기 구성

1. 연주 공정의 제어상태

연주 공정의 주조 시작시 수동제어는 조업자가 제어박스(control box)를 들고 슬라이딩 게이트(Sliding Gate, 이하 S/G으로 표기함)의 개도(opening range)를 조작하여 주형 내에 유입하는 용강의 양을 조절한다. 이리하여 주형내 용강의 레벨을 설정치(-100 mm)에 도달시킨 후 어느 정도 유지시킨 다음 정상상태 제어인 PID 자동제어로 절환한다. 주조속도 제어는 PID 자동제어로 이루어지고 있으나, 주조속도의 설정치는 숙련조업자가 여러 번의 스텝(step)의 형태로 설정치를 주고 있다. 레벨제어는 주형의 상단을 기준으로 하여 레벨이 -550mm에서 -170mm까지는 개루프제어(Open Loop Control)를 하고, 레벨이 센서에 의하여 감지되는 -170mm부터 설정치 도달까지는 퍼지 제어(FLC, Fuzzy Logic Control)를 한다. 주조속도 제어는 레벨이 -150mm지점부터 모터를 구동하여 퍼지제어로 한다. 기존의 시스템은 레벨제어와 주조 속도 제어가 독립되어 있으나, 이를 동시에 제어하려는 것이 본 연구의 목적이다.

주형 레벨제어와 주조 속도제어 루프에 관련된 전체적인 블럭도는 (그림 4)와 같다.

2. 정상상태로의 전환방법

주조가 시작된 후 주형내 용강의 레벨이 설정치에 도달하고 어느 정도 설정치를 유지하여 레벨이 안정화되면, 정상상태인 PID 자동제어로 전환한다. 이때 전환조건은 레벨의 오차가 10mm이내에 들어오고 이 상태가 10초간(연주 공장에 따라서 조금 차이는 있음) 유지되면 조업자는 PID 제어로 전환한다. 마찬가지로 주조 시작시 퍼지제어기에서 오차가 이 범위 내로 들어 오고 이 상태가 10초이상 유지되면 PID 제어로 전

환한다. 본 연구에서는 과도상태만을 다루고 정상상태는 다루지 않았다.

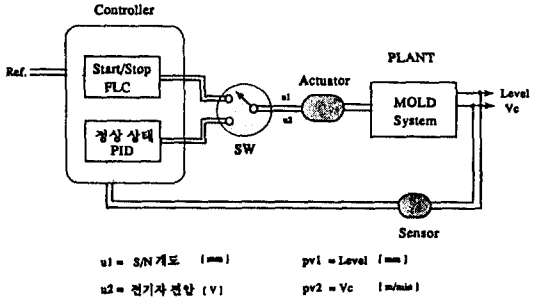

그림 4. 주형제어 루프 전체 블럭도
Fig. 4. Mold Level Control Loop block diagram.

주형 레벨의 설정치는 -100mm이다. 아래에 수식으로 정상상태로의 전환조건을 나타내었다.

$$\begin{aligned} & |\,Ref_{Level} - PV_{Level}\,| \le 10 \ [\ mm] \\ \text{and}\quad & Time[\ |\,Ref_{Level} - PV_{Level}\,| \le 10\ ] \ge 10\ [\ sec] \\ & Ref_{Level} = -100\ [\ mm] \\ & PV\_Level = \text{Sensing 되는 Level값} \end{aligned} \quad (1)$$

3. 연주 공정의 제어규칙

퍼지제어기의 설계에 있어서 규칙을 만들기 위하여 현장으로부터 조업자들과 대화와 조업 데이타를 통하여 제어규칙을 얻었다. 주형 레벨제어는 조업자들이 제어를 하고 있으므로 규칙을 얻을 수 있었지만, 주조 속도제어는 조업자들이 제어를 하고 있는 것이 아니라 설정치를 주는 형태이므로 조업자들로부터 규칙을 얻을 수 없었다. 따라서 퍼지제어에서 널리 통용되고 있는 규칙에 현장 컴퓨터의 조업 데이타를 바탕으로 규칙의 튜닝하는 작업을 하였고 모의 실험을 통하여 제어가 잘됨을 확인하였다.

표 1. 추형 레벨 제어 규칙
Table 1. Mold Level Control Rules.

L / ΔL	low	high	Ref_	Ref+	Over
NB					
NM	PB	PS	PS	PS	ZO
ZO	PB	ZO	ZO	ZO	NS
PM	PS	ZO	NS	NS	NB
PB	PS				NB

* 여기서 low는 레벨의 설정치를 기준으로 많이 낮을 때, high는 조금 낮을 때, Ref_는 설정치 근방의 아래에 있을 때, Ref+는 설정치 근방의 위에 있을 때, Over는 레벨이 설정치의 위로 많이 높여 있을 때의 퍼지화된 값임.

● 주형 레벨 제어 규칙

레벨 관련 퍼지제어는 입력이 레벨과 그의 변화량이고 출력은 S/G의 개도이다.

● 주조 속도제어 규칙

주조 속도제어는 입력이 오차와 그의 변화량이고 출력은 모터의 전기자전압의 변화량이다

표 2. 추조 속도 제어 규칙
Table 2. Casting Velocity Control Rules.

E / ΔE	NB	NS	ZO	PS	PB
NB			NB		
NS			NS		
ZO	PB	NS	ZO	PS	PB
PS			NS		
PB			PB		

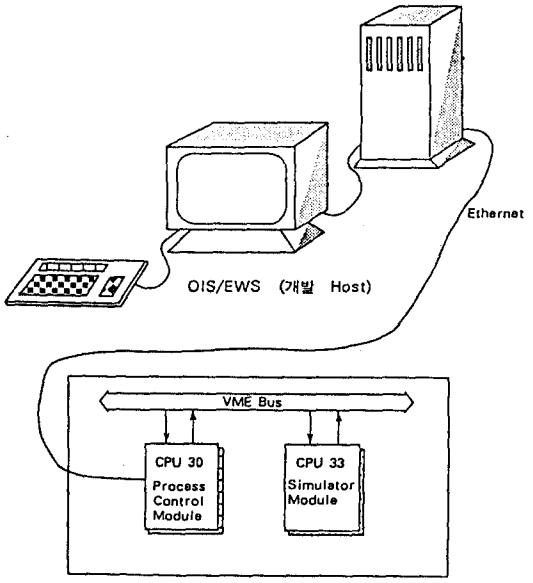

그림 5. 실험 환경
Fig. 5. Experimental environment.

V. 실험 및 결과검토

1. 기능코드에 의한 대상시스템의 실험 (기능코드의 검증)

실험환경은 (그림 5)에서 보이듯이 OIS와 EWS의 기능을 담당하는 SUN이 개발 호스트(Host)이다. 여기서 기능코드와 대상모델이 컴파일(compile)되어서 이써네트(ethernet)를 통하여 CPU30에 다운로드

(download)된다. CPU33은 VME Bus를 통하여 CPU30으로부터 대상모델을 다운로드 받는다. CPU30은 퍼지제어 기능코드가 다운로드되어 운용되는 제어부이고, CPU33은 대상모델이 운용되는 시뮬레이터부가 된다. CPU30과 CPU33은 VME Bus를 통하여 데이타(data)를 통신하며 실시간(real time)으로 운용된다. 여기서 CPU30과 CPU33이 PCS 역할을 한다.

대상모델을 제어하기 위한 퍼지제어 기능코드가 포함되어 있는 컨피그레이션 파일(configuration file)을 (그림 6)에 소개한다.

Configuration File

```
! configuration file
! FLC Controller

INITIALIZE
1 errV = msc(: 0.0001)
2 derrV = msc(: 0.0001)
3 uOLC = msc(: 75.0)
CONTROL
4 L = ain(: 0, 0, 0)
5 V = ain(: 0, 0, 1)
6 setL = msc(: 800.0)
7 setsub = msc(: 750.0)
8 setV = refv(L, setsub; )
9 dL = sumII(L, Ldelay; 1.0, -1.0)   ! dL = L(k) - L(k-1)
10 Ldelay = delay(L; 347.0)          ! Ld = L(k-1)
11 errV = sumII(setV, V; 1.0, -1.0)  ! errV = e(k)
12 derrV = sumII(errV, errdelay; 1.0, -1.0)   ! derrV =
                                              e(k) - e(k-1)
13 errdelay = delay(errV; 0.0)       ! errd = e(k-1)
! FLC of motor velocity
14 duV = flc(errV, derrV, 5.3, 5.3, 5.5,3; "RuleV.dat" )
15 minsub = msc(: 75.0)
16 minU = lmtlo(uVdelay, minsub; )   ! minU = MAX
                                              (u(k-1), 75.0)
17 uV = sumII(minU, duV; 1.0, 1.0) ! uV = u(k-1) + du
18 uVdelay = delay(uV; 75.0)         ! uVd = u(k-1)
19 nouV = msc(: 0.0)                 ! nouV = 0.0
20 lselect = msc(: 750.0)
21 coninV = select(L, lselect, uV, nouV; ) ! coninV = uV,
                                              if L > 750.0
22 olcs = msc(: 75.0)
23 uOLC = olc(olcs; )              ! uOLC = 75.0 +- 0.005
! FLC of mold Level
24 uFLC = flc(L, dL, 5.3, 5.3, 5.5,3 ; "RuleL.dat" )
25 uselect = msc(: 730.0)
26 coninL = select(L, uselect, uFLC, uOLC; ) ! coninL =
                                              uFLC, if L >730.0
! aout
27 uofL = aout(coninL; 0, 0, 0)      ! u of mold level
28 uofV = aout(coninV; 0, 0, 1)      ! u of motor speed
29 refvc = aout(setV; 0, 0, 2)       ! Ref. of motor speed
END
```

그림 6. Configuration File
Fig. 6. Configuration File.

퍼지제어 기능코드로 실험의 결과는 (그림 7)에 나타내었다. (그림 7)의 윗 그림은 주조속도를 나타내는데, 실선은 설정치를 나타내고 점선은 플랜트의 출력값을 나타낸다. 그림에서 보면 설정치를 잘 따라가고 있으므로 퍼지제어가 잘 구동하고 있음을 알 수있다. 실험에 사용된 운영체계(OS)는 Realtime OS인 Vx Works를 사용하여 실시간으로 실험하였다. 아래 그림은 주형레벨을 나타내는데 그림의 800은 설정치를 나타내고 정상상태로 제어가 잘되므로 퍼지제어 기능코드가 실시간으로 제대로 동작하고 있음을 알 수 있다. 그러므로 구현된 퍼지제어 기능코드의 검증이 되었다.

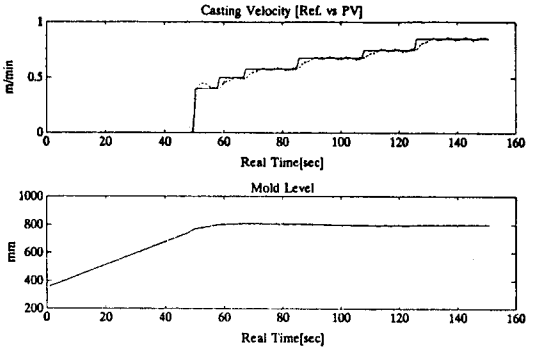

그림 7. 퍼지제어 기능코드에 의한 연주 공정의 모의 실험
Fig. 7. Continuous Casting Process simulation by Fuzzy Control Function Code.

VI. 결 론

본 논문에서는 산업체에서 운용되고 있는 DCS에 한층 진보된 제어 알고리즘인 퍼지제어 알고리즘을 구현하는 방법을 제안하였다. 본 연구에서는 퍼지제어를 현장의 사용자가 사용하기 편리하도록 기능코드의 형태로 구현하였다. 제어에 사용되는 규칙(Rule)은 데이타 파일의 형태로 읽어 들이므로 사용자가 규칙을 변경하거나 추가하려 할 때 쉽게 할 수 있게 하였다. 구현된 기능코드는 범용성을 가지므로 제철소뿐만 아니라 발전소 등의 여러 공정에도 적용이 가능하다. 구현된 기능코드의 검증을 위하여 제철소의 연주 공정을 대상으로 하여 퍼지제어를 수행하였다. 연주공정의 주조시의 수동운전을 대신할 퍼지제어기를 구성하였고 정상상태의 PID 제어기와의 상호 절환방법을 소개하였다. 제어에 사용된 규칙은 현장 조업자들로부터 얻었다. 대상시스템을 실시간 운영체제(Realtime Operating Sys-

tem)인 VxWorks로 실시간을 이용한 모의 실험을 통하여 구현된 기능코드를 검증하여 기능코드화된 퍼지 제어의 현장에 적용가능성을 보였다. 본 논문은 DCS에 퍼지제어 알고리즘의 구현하는 방법을 제시하여 실제 산업체에 적용가능성을 확인하였다는 점에서 의의가 있다고 생각한다.

앞으로 연구해야 될 과제로서는 보다 다양하고 다기능의 고급 DCS를 위하여 여러가지 고급제어 알고리즘이 구현되어야 할 것이다. 이를 위하여 신경회로망이나 적응제어, 고장진단 및 고장발생시 대처하는 기능 등의 보다 진보된 제어 알고리즘이 기능코드 형태로 구현되어야 할 것이다.

참 고 문 헌

[1] 이희규,문봉채,김병국,변증남 "그래픽 제어 언어를 사용한 공정제어용 다중루프 제어기의 개발" 한국자동제어학술회의논문집 1990. 10. 26-27 (pp. 200 - 203.)

[2] 변증남, 김병국, 박동조, 발전소 보일러 제어 시스템용 고급제어 알고리즘 및 고장진단 퍼지 전문가 시스템의 개발보고서 , 한국과학기술원, 1993. 10 (pp. 1 - 3.)

[3] 김병국, 황동환 외 6명, 분산 제어 시스템의 고장 대처 방안 및 제어 언어 구현 보고서, 한국과학기술원, 1992. 2

[4] 포항 종합 제철(주) 광양 제철소 설비관리부 정비기술과, 기술검토서 (연주공장 Tundish Sliding Nozzle 작동검토)

[5] 住友金屬工業 製鐵技術研究所, "スライテイソクノスルの油壓制御技術の開發", CAMP - ISIJ Vol 3(1990) - 1125

[6] 川崎製鐵 技術研究所本部 千葉製鐵所, "外亂オフサーハを用いた連鑄モールト內湯面 レヘル 制御", SICE'91 July 17-19 Yonezawa

[7] 포항 종합 제철(주), 철강제품과 생산 공정, (p 401 - 441)

[8] 포항 종합 제철(주) 광양 제철소, 제철 용어집, 1990

[9] 포항 종합 제철(주) 광양 제철소, 연주공장 설비 사양서, (pp. 11 - 150.)

[10] 포항 종합 제철(주) 광양 제철소, ECLM Sensor 교체 보고서

저 자 소 개

許 倫 紀(正會員)

1969年 3月 7日生. 1992年 2月 부산대학교 공과대학 전기공학과 졸업(공학사). 1994年 2月 한국과학기술원 전기 및 전자공학과 졸업(공학석사). 1992年 3月 ~ 1994년 3月 포항제철(주). 1994年 3月 ~ 1994年 6月 산업과학기술연구소. 1994年 7月 ~ 현재 포항제철(주) 기술연구소 선임연구원. 주관심 분야는 퍼지제어, 신경망제어, 인공지능의 산업계응용, 고장진단, 실시간 제어 등임.

朴 世 華(正會員) 第 32 卷 第 8 號 參照
현재 생산기술연구원 선임연구원

李 在 爀(正會員) 第 32 卷 第 8 號 參照
현재 한국외국어대학교 전임강사

卞 增 男(正會員) 第 27 卷 第 1 號 參照
현재 한국과학기술원 전기 및 전자공학과 교수