'93 한국자동제어학술회의논문집 (1993. 10. 20~22)

고급분산 제어 시스템을 위한 일반형 예측 제어 알고리즘의 개발

°김성우, 박세화, 서창준, 김병국, 박동조, 변증남
한국과학기술원 전기 및 전자공학과

Development of GPC Algorithm for the Advanced Control System

°Seong Woo Kim, Se Hwa Park, Chang Joon Seo
Byung Kook Kim, Dong Jo Park and Zeungnam Bien
Dept. of Electrical Engineering, KAIST

Abstract

In this paper, the GPC algorithm is developed for ACS (advanced control system). ACS equals to DCS (distributed control system) with some advanced control algorithm, for example, fuzzy logic controller, autotuning. By its embedded structural control language, which uses simple function codes corresponding to each function blocks, it is possible to construct multiloop controller. The developed GPC function code is divided by RLS (recursive least square) parameter estimator and GPC controller. Simulation result show the availability of GPC fuction code using the control language.

1 서 론

80년대에 등장하기 시작한 분산제어 시스템 (Distributed Control System) 은 마이크로프로세서 기술의 발달이 산업 공정제어 분야에 영향을 끼쳐 공정 제어의 자동화, 모듈화가 가속된 결과의 소산이다. 그러나, 공정 시스템의 제어 성능 향상을 위해서는 PID 제어기 같은 종래의 고전적인 제어기들로 구성된 기존의 분산제어 시스템에 현대 제어 이론을 바탕으로 한 고급 제어 알고리즘들을 장착한 고급 분산제어 시스템 (Advanced Control System) 이 필연적이다.

분산 제어 시스템의 기본 구성은 크게 플랜트 제어를 담당하는 PCS (Process Control System), engineering 작업을 하는 EWS (Engineering Workstation), PCS에서 보낸 공정 정보 및 제어값들을 그래픽 처리로 보여주는 OIS (Operator Interface System)로 나누어진다. 일반적으로 시스템 제어 프로그램은 EWS에서 작성되어 PCS로 옮겨져 실행되는데, 그 효율성, 신뢰성은 소프트웨어(제어 언어)에 의해 좌우된다. 본 논문에서는 공정제어에 적합한 기능 블럭 개념을 도입한 이미 개발된 고급 제어 언어를 사용하였다[1, 2, 3].

한편, 대부분의 산업 공정 시스템은 미래의 시스템 정보를 이용하게 되는데, 현대 제어 이론을 기반으로 한 진보된 제어 알고리즘 중 이에 가장 부합하는 것이 모델 예측 제어이고 실제로 많은 산업용 제어 부문에서 사용되고 있다[4, 6]. 기존의 모델 예측 제어 알고리즘들의 장점만을 취한 일반형 예측 제어 (Generalized Predictive Control)는 비최소위상, 불안정, 시간지연을 갖는 공정 등에 모두 적합한 보편적 방식으로 알려져 있다[5]. 그러나, 대규모 분산 제어 시스템에 쉽게 적용할 수 있게 구현된 예는 아직 국내에는 알려진 바가 없다.

본 논문에서는 일반형 예측 제어 알고리즘을 이미 개발된 제어 언어를 이용하여 구현하여 고급 분산제어 시스템에 쉽게 적용할 수 있다는 것을 보여준다. 또한, 화력발전소 보일러 시뮬레이터에 적용하여 그 유용성을 확인하였다.

논문의 구성은 1장 서론에 이어 2장에서 개발된 제어 언어에 대해 간단히 설명하고, 3장에서는 예측 제어 기능코드에 대해 기술되고, 4장에서 시뮬레이션 사례를 보이고 5장에서 결론을 맺는다.

2 구조적인 제어 언어

일반적인 공정 제어 알고리즘은 다입력 다출력 구조이며 내부적으로는 여러 국부 루프 (local-loop)의 복합적인 구성으로 이루어진 다중 루프(multi-loop) 제어기이다. 다중루프 제어기 같은 복잡한 제어 시스템 구성을 용이하게 하기 위해서 모듈화에 의한 확장성이 필요하다. 이것을 가능하게 하는 것이 configurable controller이다.

Configurable controller는 기능 블럭(function block) 개념의 언어로 구성하는데, 이를테면, Baily에서 구현한 형식이다[3]. 이 기능 블럭들은 제어기 블럭선도에서 각각의 블럭들에 대응하는 기능들로써, 자주 사용되고 유용한 기능들을 개발자가 기능 서브루틴(function subroutine)으로 미리 구현해 놓게 된다. 사용자는 이들을 유기적으로 결합 (configuration)하여 제어기를 구성한다. 이 때 각 기능 블럭 내의 구조를 알 필요가 없으며, 단지 그 입력, 출력, 기능을 이해하면 된다. 이러한 개념에 기초하여 보면 사용된 언어는 일종의 문제 지향 언어(Problem Oriented Language)이다.

기능 블럭들은 기능 코드(function code)들로 표현된다. 기능 코드는 기능 블럭들 사이의 입출력 관계와 내재된 계수들을 사용자가 쉽게 이해할 수 있고 간단한 ASCII문자로 표현한 것이다. 따라서 제어기의 구성은 기능코드들의 나열로써 이루어질 수 있다. 기능코드의 표현방식은 다음과 같다.

블럭번호 블럭변수= 기능코드이름(입력포트들; 파라미터값들)

기능 코드들의 나열로써 이루어진 ASCII file을 configuration file이라 하며 그 예는 다음과 같다.

```
!This is the example of configuration file
 1 a = msc(;100.0)          !manual set constant
 2 b = ain(;0,1,1)          !analog input
 3 c = sumII(1,2;1.0,-1.0)!weighed summation
 4 d = aout(3;0,0,0)        !analog output
!End of example file
```

이것은 1번블럭의 출력인 100.0과 외부로부터의 입력인 2번출력의 차를 외부로 출력하는 기능을 수행한다.

기능코드로 된 configuration 파일을 타겟 프로세서가 직접 이해할 수 없으므로, 이식성이 강한 C 언어 프로그램으로 변환시킨 다음 타겟 프로세서에 맞는 cross compiler를 써서 기계어로 바꾸어 주게 된다. configuration file의 C-언어 프로그램으로으로의 변환은 UNIX 제공 유틸리티인 lex와 yacc를 이용하여 구현된다. 생성 방법은 각 기능 코드에 해당하는 함수를 기능 코드 형식에 적합한 구조로 C 함수(subroutine)로 작성하고, 기능 코드들의 순서에 따라 함수들을 호출하는 것이다.
또한 이와는 별도로, PCS에서 수행되는 통신 태스크(task), 명령어 처리 태스크에 대한 프로그램이 필요하다. 여기서, 명령어 처리 태스크는 OIS에서 받은 명령을 분석하여 거기에 맞게 처리를 해 주게 된다. 다음 그림은 기능코드 파일로부터 C 언어 프로그램이 생성되고 최종적으로 제어 태스크가 생기는 기본 구조를 보여준다.

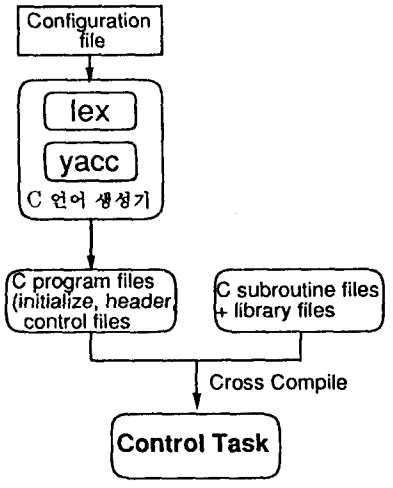

그림 1: 제어 태스크 생성의 전체 흐름도

3 예측 제어 기능 코드

일반형 예측 제어기는 계수 추정기와 제어기가 분리된 간접(indirect) 적응 제어기이다[5]. 그러므로, 기능 코드로 구현할 때도 계수 추정기와 제어기를 따로 분리하는 것이 타당하다. 다음 그림 2는 GPC 제어기의 블럭선도이다.

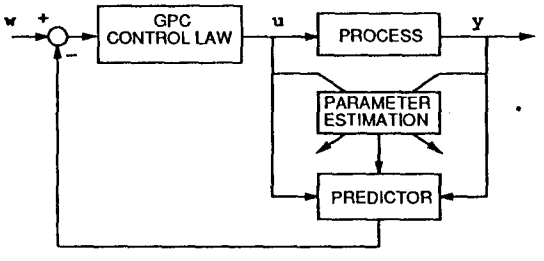

그림 2: GPC 페루프 시스템

본 논문에서는 계수 추정법으로 일반적으로 많이 쓰이는 RLS (Recursive Least Square) 알고리즘을 기능코드로 구현하고, GPC 제어기도 따로 기능코드로 구현하는 방법을 채택하였다. 결국, 실제 사용시에는 계수 추정기와 제어기 기능코드 2개를 연결하여 하나의 완성된 GPC 기능코드를 구성하게 된다. 그 구성도는 다음 그림 3과 같다. 또한, 플랜트 차수에 따라 1차, 2차, n차의 기능코드를 따로 구현하여 일반화하였다.

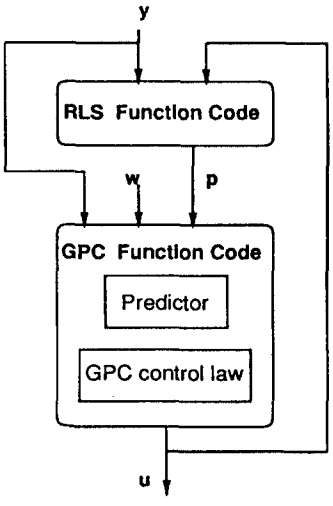

그림 3: GPC 기능코드 구성도

3.1 RLS 기능코드

RLS 기능코드의 형식은 다음과 같다.

p = rlsi (y, u, trsw, delay, order ; ...)

i : 플랜트 차수 (1,2,n)
p : 추정 계수 벡터
y : 공정 출력
u : 제어 입력
trsw: Track Switch(1:정상동작, 0:no operation, others:에러 발생)
delay: 공정 지연 시간(process delay time)
order: 공정 차수(n과 m, rlsn일때만 적용)

RLS 기능코드는 제어 입력, 공정 출력의 플랜트 정보를 가지고 플랜트 계수를 추정하는 부분이다[7]. 플랜트 차수에 따라 1차, 2차, 임의의 차수로 구현하였는데, 지연시간이 있는 일반적인 경우까지 고려하였다. MA부분은 시스템이 causal인 걸 감안하면 1차, 2차의 경우 각각 $\leq 1, \leq 2$가 되나, 임의의 차수의 경우 따로 order입력을 두어 AR, MA 차수를 설정한다. 또한, 기본적으로 forgetting factor를 파라미터로 가진다.

3.2 GPC 기능코드

GPC 기능코드의 형식은 다음과 같다.

u = gpci (w, y, p, ytr, trsw, delay, order, N, Nu ; ...)

i : 플랜트 차수
u : 최종적으로 계산된 제어 입력
w : 예측 가능한 기준 입력
y : 공정 출력
p : 추정 계수 벡터
ytr: Track Signal
trsw: Track Switch(1:정상동작, 0:ytr을 출력, others:에러 발생)
delay: 공정 지연 시간(process delay time)
order: 공정 차수(n과 m, gpcn일때만 적용)
N : 최대 출력 예측 구간
Nu: 제어 구간

여기서는 RLS기능코드에서 추정된 시스템 계수들을 받아서 일반형 예측제어 법칙을 통해 최종 제어 입력을 계산해내는 부분이다. 특별히, track signal과 switch를 두어 다른 제어기를 사용할 경우 trsw를 0으로 하여 선택기능을 강화하였다.
또, RLS, 예측 제어 기능코드에서 입력들(delay,order,N,Nu 등)에 따라 내부 연산 행렬들의 차원이 가변인 점을 고려하여, C 언어 프로그램 생성 시 초기 메모리 할당 루틴을 따로 만들어 내도록 하였다.

구현된 기능코드는 다음 표 1과 같다.

기능코드 이름	내용
rls1	1차 플랜트의 계수 추정
rls2	2차 플랜트의 계수 추정
rlsn	임의의 차수 플랜트의 계수 추정
gpc1	1차 플랜트의 예측 제어
gpc2	2차 플랜트의 예측 제어
gpcn	임의의 차수 플랜트의 예측 제어

표 1: 구현된 기능 코드들

4 실 험

4.1 실험 환경

예측 제어 기능코드의 실험을 위해 서울화력 4호기 보일러를 모사하는 시뮬레이터를 사용하였다[1, 2]. 이 모델에서 연소제어, 노내압 제어, 증기 온도 제어, 드럼 수위 제어의 주요 4개 드럼 보일러 제어 루프 중 급수 제어기에 해당하는 드럼 수위 제어 루프에만 예측 제어기로 적용하였다. 여기서, 다른 제어 루프는 일반적인 PID 제어를 하게 된다. 사용된 시뮬레이션 시스템은 다음과 같이 구성된다.

OIU&EWS : SUN Workstation with UNIX
PCS : VME system, VxWorks(real time OS)
CPU30 microprocessor : controller
CPU30 microprocessor : simulator
communication : Ethernet

4.2 실험 방법

시뮬레이터는 이미 구현되어 있다고 가정할 때, 제어기 실험 순서는 다음과 같다.

1. GPC 기능코드를 이용해서 서울화력 4호기 보일러 시뮬레이터에 대한 제어 알고리즘(configuration file)을 구성한다.
2. C 언어 프로그램으로 변환한다.
3. 생성된 C 언어 프로그램들을 타겟 프로세서에서 실행할 수 있도록 compile한다. (GNU cross compiler 사용)
4. 실행파일을 네트웍 통신을 통해 타겟 프로세서에 download한다.
5. 시뮬레이터를 실행하고 제어기를 실행한다.

다음은 실제로 구현한 드럼 수위 GPC 제어에 대한 기능코드파일이다.

```
! Drum Level Loop
! GPC Controller
56 L= ain( ; 0, 2, 4) !-------------------L
57 offset1= msc( ; 50.0)
58 y= sumII(L, offset1; 1.0, -1.0)
59 offset2= msc( ; 0.0)
60 u= sumII(u1, offset2; 1.0, 0.0)
61 trsw= mss( ; 1)
62 p= rlsn(y, u, trsw, 0, 4, 4; 0.0, 0, 0.9)
63 w= msc( ; 0.0)
64 track= msc( ; 0.0)
65 u1= gpcn(w,y,p,track,trsw,0,4,4,10,1;
0.0,-50.0,50.0,0.1)
66 offset3= msc( ; 60.0)
67 bfp1= sumII(u1, offset3; 0.5, 1.0)
68 bfp2= hilo(bfp1; 100.0, 0.0)
69 offset4= msc( ; 100.0)
70 bfp3= sumII(bfp2, offset4; 1.0, -1.0)
71 bfp4= hilo(bfp3; 100.0, 0.0)
72 bfpa= aout(bfp2; 0, 1, 4) !----------BFPa
73 bfpb= aout(bfp2; 0, 1, 5) !----------BFPb
74 bfpc= aout(bfp4; 0, 2, 0) !----------BFPc
```

4.3 실험 결과

다음 그림들은 실제로 위의 configuration file을 이용해서 ACS를 시뮬레이션한 결과를 나타낸다. 여기서 제어 목적은 드럼 수위를 일정하게 유지(50%)하는 것이다. 결과에서 알 수 있듯이 구현된 GPC 제어기가 PCS에서 무리없이 잘 동작되고 있다. 초기의 나쁜 특성은 전체 시스템의 갑작스런 펌프 제어와 RLS에서 시스템 계수를 충분히 추정하지 못한 데 따른 예측 제어의 영향을 나타낸다.

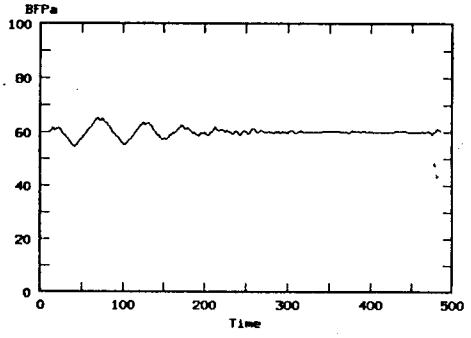

그림 4. 제어 입력(Boiler Feedwater Pump) 결과

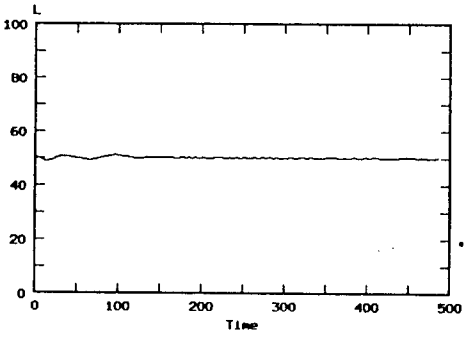

그림 5. 공정 출력(Drum Level) 결과

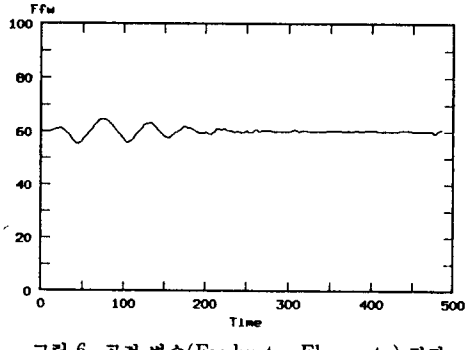

그림 6. 공정 변수(Feedwater Flow-rate) 결과

5 결 론

본 논문에서 고급 분산 제어 시스템에 적용 가능하도록 RLS 계수 추정을 포함한 예측 제어 알고리즘을 기능코드로 구현하였다. 그리고, 실제로 고급 분산 제어 시스템에서 실험함으로써 그 유용성을 증명하였다. 구현된 기능코드들은 일반적인 단입력단출력 플랜트에 대해서 적용할 수 있으며, C 언어로 구현되어서 여러 공정 제어 시스템에 사용할 수 있다. 고속 프로세서에 의한 실시간 제어가 점차 이루어지므로, 앞으로 다입력 다출력 예측 제어 알고리즘을 비롯한 여러 변형된 고급 알고리즘들의 기능코드 구현이 이루어질 것으로 사료된다.

참고 문헌

[1] 한전 기술 연구원, "발전소 제어용 계장 제어 시스템 개발(최종 보고서)", 1990. 8.

[2] 한전 기술 연구원, "분산 제어 시스템의 고장 대처 기능 및 제어 언어의 구현(최종 보고서)", 1993. 3.

[3] Baily Controls, *Baily Network 90 - Fuction Code Reference Manual*,

[4] D. W. Clarke, "Application of Generalized Predictive Contol to Industrial Processes", *IEEE Contr. Sys. Magazine*, pp49-55, April, 1988

[5] D. W. Clarke, C. Mohtadi and P. S. Tuffs, "Generalized Predictive Control-Part I. The Basic Algorithm", *Automatica*, vol. 23, No. 2, pp 137-148, 1987

[6] J.A. Rossiter, B. Kouvaritakis and R.M. Dunnett, "Application of generalized predictive contol to a boiler-turbine unit for electricity generation", *IEE Proceedings-D*, vol. 138, No. 1, pp 59-67, Jan, 1991

[7] G.C. Goodwin and Kwai Sang Sin,*Adaptive Filtering Prediction And Control*, Prentice-Hall, 1984