ファジィ制御用プログラミング言語の開発とその応用*

古川 万寿夫** ・三浦 健史*** ・松田 孝史****

（平成6年10月26日 受理）

The Programming Language for Fuzzy Control and It's Application

By Masuo Furukawa, Takeshi Miura and Takashi Matsuda

We designed the programming language for fuzzy control using micro processor. This language permits to describe if-then rule into the program for fuzzy control. This language is easy to understand for fuzzy control engineer. We developed the Fuzzy to C translator which translates this programming language into C language. The program generated by the translator is compiled by C compiler to apply to target for fuzzy control.

1. はじめに

1965年にカリフォルニア大学のザデー教授（L. A. Zadeh）によって，「ファジィ集合」が提唱された(1)．ファジィ集合は境界があいまいな集合であり，人間の主観的判断やあいまいさを定量的に取り扱うために提案された．

当時は，科学に主観性を導入したとして批判されたが，1970年代後半から人工知能やエキスパートシステムの研究がさかんになり，現実問題に含まれる主観性やあいまいさにも目が向けられるようになってきた．1980年にはデンマークのシュミット社によるセメントキルンのファジィ制御が実用化され(2)，1987年には仙台の地下鉄の自動運転にファジィ制御が用いられる(3)など，ファジィ制御の有用性が示された．

ファジィ制御は「もし，制御対象XがAの状態ならば，出力YをBにせよ．」のような，直感的に理解ができるif～then…ルールを用いて行う制御である．現在，ほとんどのファジィ制御は，マイクロプロセッサを用いて行われている．ファジィ制御の長所のひとつは，制御則を直感的に理解できることである．ところが，ファジィ制御をC言語などの汎用プログラミング言語で実現すると，プログラムを一目見ただけでは制御則を直感的に理解できないこと，汎用言語に関する知識が必要となること，ファジィ推論のアルゴリズムをプログラムにすべて記述しなければならないので開発に時間がかかることなどの不利な点が

---

* 平成6年6月第10回ファジィシステムシンポジウムにて発表

** 電気工学科 講師

*** 京都大学工学部 在学中

**** 九州工業大学工学部 在学中

ある.

もし，if～then…ルール形式でプログラミングできる言語があれば，ファジィ制御をおこなうためのプログラムを短時間で開発することができ，かつ直感的に理解しやすいプログラムを作ることができる.

本研究では，マイクロプロセッサによるファジィ制御のプログラムをif～then…ルールで記述できるファジィ制御用プログラミング言語を設計した．このプログラミング言語で記述したソースプログラムをC言語に変換するトランスレータをｙａｃｃを用いて試作した．そして，トランスレータの出力したC言語ソースプログラムを市販のCコンパイラでコンパイルし，オブジェクトプログラムを生成することができた.

2. マイクロプロセッサを用いたファジィ制御

現在，ほとんどのファジィ制御は，マイクロプロセッサを用いて行われている．ファジィ制御をマイクロプロセッサを用いておこなう場合，C言語などの汎用言語を用いてプログラミングをする.

一般にパソコンでは，実行プログラムをハードディスクもしくはフロッピーディスクで起動し制御をおこなう．ワンボードマイコンの場合は，実行プログラムをROM化して制御を実現する．このようにパソコンとワンボードマイコンではプログラムの格納方法が異なるが，パソコンやワンボードマイコンを用いた制御はマイクロプロセッサを用いた制御であるという点では同じである.

マイクロプロセッサを用いたファジィ制御のインターフェイスは，A／D変換器とD／A変換器が用いられることが多い．A／D変換器は，センサからのデータをディジタル値に変換してマイクロプロセッサへ入力する．一方，D／A変換器は，マイクロプロセッサからの出力をアナログ値に変換して出力する.

マイクロプロセッサは，A／D変換器からのデータを入力し，ファジィ推論をおこなう．そして，ファジィ推論により決められた制御量をD／A変換器へ出力し，アクチュエータを制御する.

このことから，ファジィ制御をおこなうプログラムは，ファジィ推論に必要なif～then…ルールやメンバーシップ関数の設定だけでなく，A／D変換器とD／A変換器の最大入出力値，そしてI/0ポートのアドレスなどのインターフェイスのパラメータも設定できなければならない.

本プログラミング言語は，if～then…ルールとメンバーシップ関数のほかに，最大入出力値およびI/0ポートのアドレスの設定ができる.

3. ファジィ制御用プログラミング言語

3-1 仕 様

以下に，本プログラミング言語の仕様を示す.

- 推論方式はMAX-MIN合成法
- 非ファジィ化は面積重心法
- メンバーシップ関数は三角型が使用できる
- 入出力の変数は合わせて最大５個

表1 予約語

入出力設定部	ﾒﾝﾊﾞｰｼｯﾌﾟ関数設定部	if～then…ルール設定部
input	membership	rule
output	label	if
fs	left	then
port	center	is
	right	and

まで
- ラベルの数は最大７個まで
- 予約語とまったく同じ名前の変数やラベルは使えない
- 変数やラベルの長さは最長３２文字まで
- 表１に予約語を示す
- ”/*”と”*/”で囲むことで注釈行を記述できる

3-2 記述形式

プログラム記述形式の説明のため，気温を入力とし，ダイアル値を出力とするエアコンのファジィ制御を考えることにする．

図１にこのファジィ制御を本プログラミング言語で記述した１入力１出力ファジィ制御のソースプログラムを示す．本プログラミング言語のソースプログラムは，入出力設定部，メンバーシップ関数設定部およびif～then…ルール設定部の三つで構成する．以下に，各設定部の記述形式について述べる．

（1）入出力設定部

入出力設定部では，入出力変数の定義，入出力値の最大値の設定および入出力ポートのアドレスの設定をおこなう．図１の入出力設定部は，入力変数名がtemperature，出力変数名がdialになっている．

ここで用いている予約語は，”input”，”output”，”fs”，”port”の四つである．”input”は入力の設定部であることを，”output”は出力の設定部であることを宣言する．”fs”は，”full scale”の略で，入出力値の最大値を設定する．”port”は，入出力ポートのアドレスを指定する．

```
/* 入出力設定部 */
input {
    temperature  :fs=40 , port=0x70d6 ;
}
output{
    dial :fs=20 , port=0x70d0 ;
}
/*メンバーシップ関数設定部*/
membership {
    input [
        temperature (
        label=PB :left=34,center=40,right=40 ;
        label=PM :left=27,center=34,right=40 ;
        label=PS :left=20,center=27,right=34 ;
        label=ZR :left=13,center=20,right=27 ;
        label=NS :left= 6,center=13,right=20 ;
        label=NM :left= 0,center= 6,right=13 ;
        label=NB :left= 0,center= 0,right= 6 ;)
    ]
    output[
        dial (
        label=PB :left=16,center=20,right=20 ;
        label=PM :left=13,center=16,right=20 ;
        label=PS :left=10,center=13,right=16 ;
        label=ZR :left= 7,center=10,right=13 ;
        label=NS :left= 4,center= 7,right=10 ;
        label=NM :left= 0,center= 4,right= 7 ;
        label=NB :left= 0,center= 0,right= 4 ;)
          ]
}
/*if-thenルール設定部*/
rule {
    if  temperature is PB then dial is NB ;
    if  temperature is PM then dial is NM ;
    if  temperature is PS then dial is NS ;
    if  temperature is ZR then dial is ZR ;
    if  temperature is NS then dial is PS ;
    if  temperature is NM then dial is PM ;
    if  temperature is NB then dial is PB ;
}
```

図１ エアコンのファジィ制御プログラム

（2）メンバーシップ関数設定部

メンバーシップ関数設定部では，ラベルの定義と各ラベルに対するメンバーシップ関数の形状を記述する．図１のメンバーシップ関数設定部は，入出力設定部で定義したtemperature，dialの変数に対してそれぞれ７つのメンバーシップ関数を割り付けた場合である．

ここでは，”membership”，”label”，”left”，”center”，”right”という五つの予約語が使われている．”membership”は，メンバーシップ関数の設定部を明示するための宣言である．”label”はラベルの定義をし，”left”，”center”そして”right”は，三角型メンバーシップ関数の形状を設定する．

（3）if～then…ルール設定部

if～then…ルール設定部では，ファジィ制御の制御則をif～then…形式で記述する．

ここで使われる予約語は，”rule”，”if”，”then”，”is”，”and”の五つである．”rule”はif～then…ルール設定部であることを明示するための宣言である．”if”は制御則の始まりを，”then”は前件部と後件部の区切りをあらわす．”and”は前件部や後件部に複数のand条件を列挙するときに用いる．

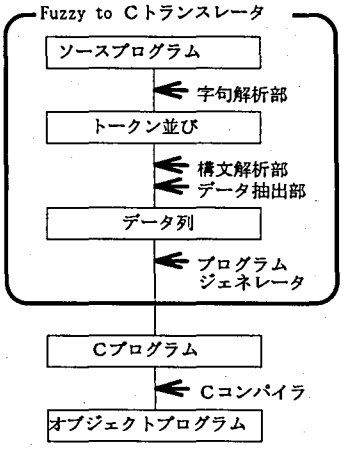

図2 Fuzzy to Cトランスレータ

4: トランスレータ Fuzzy to C

本研究では，本プログラミング言語をC言語に変換するトランスレータFuzzy to Cを試作した．このトランスレータの出力するC言語プログラムを市販のCコンパイラでオブジェクトプログラムに翻訳し，実機制御に用いた．ここでは，試作したFuzzy to Cトランスレータについて述べる．

トランスレータの構成を図2に示す．Fuzzy to Cトランスレータは，字句解析部，構文解析部，データ抽出部そしてプログラムジェネレータの四つからできている．

（1）字句解析部

字句解析部は，ソースプログラムから予約語や変数，数字などを分離しトークンに変換する．

（2）構文解析部

構文解析部は，ソースプログラムを解析

```
#include   <stdio.h>
#define    _dx_       1
#define    _tmax_     40.0
#define    _tmin_     0.0
#define    _cmax_     20.0
#define    _cmin_     0.0
#define    _nomf_     7
typedef struct{
    int  min,
         center,
         max;
}member;
double Mvalue(member *data, double x)
{       double a,b,c,center;
        int    max,min;
        if(x < data->min || data->max <x)
            return (0.0);
        center = data->center;
        max=data->max;
        min=data->min;
        if(x>=center)
            return ((max-x)/(max-center));
        else
            return ((x-min)/(center-min));
}
void main(void)
{
    member mem1[_nomf_] = {
                          {34,40,41},
                          {27,34,40},
                          {20,27,34},
                          {13,20,27},
                          { 6,13,20},
                          { 0, 6,13},
                          {-1, 0, 6}};
    member mem2[_nomf_] = {
                          {16,20,21},
                          {13,16,20},
                          {10,13,16},
                          { 7,10,13},
                          { 4, 7,10},
                          { 0, 4, 7},
                          {-1, 0, 4}};
    double  grade[_nomf_];
    double  tem,max,min,area,d_area,
            moment,center,work1,work2,work3;
    int     i;
    for(tem=_tmin_;tem<=_tmax_;tem++) {
        for(i=0;i<_nomf_;i++)
            grade[i] = Mvalue(&mem1[i],tem);
        area=0.0;
        moment=0.0;
        for(work1=_cmin_;work1<=_cmax_;work1+=_dx_){
            max = 0;
            for(i=0;i<_nomf_;i++){
                work2=Mvalue(&mem2[i],work1);
                if(work2<grade[i])
                    work3=work2;
                else
                    work3=grade[i];
                if(work3>max)
                    max=work3;
            }
            d_area=max;
            area+=d_area;
            moment+=d_area*work1;
        }
        if(area!=0)
            center= moment/area;
        printf("%d:%lf¥n", (int)tem,center);
    }
}
```

図3 エアコンファジィ制御のCプログラム

して文法に合致しているか調べる.

（3）データ抽出部

データ抽出部は，プログラムジェネレータがCプログラムを生成するのに必要なデータを，構造体や配列に階層的に順序よく格納する.

（4）プログラムジェネレータ

プログラムジェネレータは，制御用のCプログラムを作り出す部分である.

5. 汎用言語との比較

ファジィ制御をおこなうプログラムを汎用言語であるC言語で記述した場合と，本プログラミング言語で記述した場合について比較評価する.

図1と同等のファジィ制御をするCプログラムを記述すると，図3のように73ステップとなった．C言語で記述したプログラムの特徴は，一目見ただけでは制御則が理解できないこと，変数が多くなるとプログラムがさらに複雑になることである．本プログラミング言語の場合は図1から37ステップであることがわかる（コメント行を除く）．本プログラミング言語では制御則が多くなるとそれだけステップ数が増大するが，if～then…形式の記述なのでC言語の場合に比べ理解しやすいと言える.

このことから本プログラミング言語を用いた方が，分かりやすいプログラムを短時間で開発することができるといえる.

6. 倒立振子制御への応用

本プログラミング言語の応用として倒立振子制御をおこなった．ここでは，倒立振子の実験装置および制御用のソースプログラムそしてその結果について述べる.

6-1 実験装置

実験のため製作した倒立振子系を，図4に示す．簡単化のため角度1入力，モータ電圧1出力のファジィ制御とした.

6-2 ソースプログラム

制御に用いたソースプログラムを図5に示す．メンバーシップ関数は，前件部・後件部ともに等間隔に割り当てられた三角型とした.

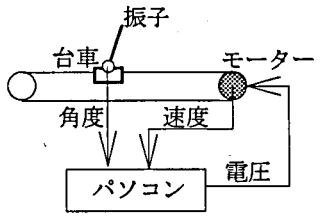

図4 倒立振子制御系

```
input {
    angle  :fs=256 , port=0x70d6 ;
}
output{
    motor  :fs=256 , port=0x70d0 ;
}
membership {
    input [
        angle (
        label=PB :left=213,center=255,right=255 ;
        label=PM :left=171,center=213,right=255 ;
        label=PS :left=128,center=171,right=213 ;
        label=ZR :left= 85,center=128,right=171 ;
        label=NS :left= 43,center= 85,right=128 ;
        label=NM :left=  0,center= 43,right= 85 ;
        label=NB :left=  0,center=  0,right= 43 ;)
    ]
    output[
        motor (
        label=PB :left=213,center=255,right=255 ;
        label=PM :left=171,center=213,right=255 ;
        label=PS :left=128,center=171,right=213 ;
        label=ZR :left= 85,center=128,right=171 ;
        label=NS :left= 43,center= 85,right=128 ;
        label=NM :left=  0,center= 43,right= 85 ;
        label=NB :left=  0,center=  0,right= 43 ;)
        ]
}
rule {
        if  angle is PB then motor is PB ;
        if  angle is PM then motor is PM ;
        if  angle is PS then motor is PS ;
        if  angle is ZR then motor is ZR ;
        if  angle is NS then motor is NS ;
        if  angle is NM then motor is NM ;
        if  angle is NB then motor is NB ;
        }
```

図5 倒立振子のファジィ制御プログラム

6-3 結 果

Fuzzy to C トランスレータが出力した倒立振子ファジィ制御のCプログラムを図6に示す．このCプログラムをTurbo C Ver 2.0でコンパイルした結果，オブジェクトプログラムの大きさは2748バイトとなった．推論速度は，PC9801RA 20MHz（CPUをCyrix製CX486DLC, FPUをCyrix製83D87に改造）を使用した場合11msであった．

このオブジェクトプログラムで実機の制御を行った．振子を手で持ちながら直立させた状態から左または右に傾けると，台車は振子を直立させるように台車は動いた．しかし，手を放すと振子の倒立状態を保つことができなかった．その原因は，入力変数が角度しか用いていないこと，メンバーシップ関数の調整ができていなかったことであると考える．

7. 結 論

本研究では，マイクロプロセッサによるファジィ制御のプログラムをif～then…ルール形式で記述できるファジィ制御用プログラミング言語を設計した．このプログラミング言語で記述したソースプログラムをC言語に変換するトランスレータFuzzy to Cをｙａｃｃを用いて試作した．そして，トランスレータの出力したC言語ソースプログラムを市販のCコンパイラでコンパイルし，オブジェクトプログラムを生成することができた．

応用として，倒立振子制御のプログラムをプログラミング言語で記述し制御を試みた．その結果，直感的にわかりやすいプログラムが記述できることがわかった．

参 考 文 献

(1) L. A. Zadeh: Fuzzy Sets, INFORMATION AND CONTROL, Vol. 8, pp. 338-353 (1965)
(2) E. H. Mamdani: Applications of Fuzzy Algorithms for Control of Simple Dynamic Plant, Proc. IEE, Vol. 121, No. 12, pp. 1585-1588 (1974)
(3) 安信誠二: ファジィ推論を利用した列車自動運転, 情報処理, Vol. 30, No. 8, pp. 970-975 (1989)

```
#include <stdio.h>
#include <dos.h>
#define     left        0
#define     center      1
#define     right       2
#define     number      3
void        value(int x1, int x2, double *a, double *b)
{
            *a=1.0/(x2-x1);
            *b=-(*a)*x1;
}
double in_val(int argument, int in_data[][4], int data)
{
            int          x;
            double       a, b;

            if(data<in_data[argument][left]||in_data[argument][right]<data)
                        return(0.0);
            x=(data<in_data[argument][center])?in_data[argument][left]:in_data[argument][right];
            value(x, in_data[argument][center], &a, &b);
            return(data*a+b);
}
void out_val(int argument, int out_data[][4], double grade, int fs, double out[][256])
{
            int          i, small, large;
            double       x1, x2, a1, a2, b1, b2;
```

```
            value(out_data[argument][left],out_data[argument][center],&a1,&b1);
            x1=(grade-b1)/a1;
            value(out_data[argument][right],out_data[argument][center],&a2,&b2);
            x2=(grade-b2)/a2;

            small=(0<out_data[argument][left])?out_data[argument][left]:0;
            large=(out_data[argument][right]<fs)?out_data[argument][right]:fs;

            for(i=small;i<large;i++){
                        if((double)i<x1){
                                    if(out[out_data[argument][number]][i]<a1*i+b1){
                                                out[out_data[argument][number]][i]=(a1*i+b1<grade)?a1*i+b1:grade;
                                    }
                        }
                        else if((double)i>=x2){
                                    if(out[out_data[argument][number]][i]<a2*i+b2){
                                                out[out_data[argument][number]][i]=(a2*i+b2<grade)?a2*i+b2:grade;
                                    }
                        }
                        else if(out[out_data[argument][number]][i]<grade)
                                    out[out_data[argument][number]][i]=grade;
            }
}
unsigned char gchar(void)
{
            _AH=6;
            _DL=0xff;
            geninterrupt(0x21);
            return _AL;
}
void main(void)
{
            static int in_func[][4]={
                                                                                    213,255,256,0,
                                                                                    171,213,255,0,
                                                                                    128,171,213,0,
                                                                                    85,128,171,0,
                                                                                    43,85,128,0,
                                                                                    0,43,85,0,
                                                                                    -1,0,43,0,
                                                                                    };
            static int out_func[][4]={
                                                                                    213,255,256,0,
                                                                                    171,213,255,0,
                                                                                    128,171,213,0,
                                                                                    85,128,171,0,
                                                                                    43,85,128,0,
                                                                                    0,43,85,0,
                                                                                    -1,0,43,0,
                                                                                    };

            static long int in_port[][2] ={
                                                                                    256,28886,
                                                                                    };
            static long int out_port[][2]={
                                                                                    256,28880,
                                                                                    };
            static int in_rule_data[]={
                                                                                    0,
                                                                                    1,
                                                                                    2,
                                                                                    3,
                                                                                    4,
                                                                                    5,
                                                                                    6,
                                                                                    };
            static int out_rule_data[]={
                                                                                    0,
                                                                                    1,
                                                                                    2,
                                                                                    3,
                                                                                    4,
                                                                                    5,
                                                                                    6,
```

```
                                                                        };
static int in_rule_count[]={
                                                                        1,
                                                                        1,
                                                                        1,
                                                                        1,
                                                                        1,
                                                                        1,
                                                                        1,
                                                                        };
static int out_rule_count[]={
                                                                        1,
                                                                        1,
                                                                        1,
                                                                        1,
                                                                        1,
                                                                        1,
                                                                        1,
                                                                        };
double grade,x,Y,XY;
double out[1][256];
int i,j,input_data,output_argument,input[1];
int in_sum,out_sum;
int output;
outportb(out_port[0][1],128);
outportb(in_port[0][1],0);
printf("push 'q' key if you want to quit¥n");
while(gchar()!='q'&&gchar()!='Q'){
        while((inportb(0x70d0)&0x08)!=0);
        input[0]=inportb(in_port[0][1]);
        if(in_port[0][0]<input[0]){
                printf("input data is out of range!");
                outportb(out_port[0][1],128);
                exit(1);
        }
        outportb(in_port[0][1],0);
        in_sum=out_sum=0;
        for(i=0;i<1;i++)
                for(j=0;j<out_port[i][0];j++)
                        out[i][j]=0;
        for(i=0;i<7;i++){
                grade=1.0;
                for(j=0;j<in_rule_count[i];j++){
                        input_data=input[in_func[in_rule_data[in_sum]][number]];
                        x=in_val(in_rule_data[in_sum++],in_func,input_data);
                        if(x<grade) grade=x;
                }
                for(j=0;j<out_rule_count[i];j++){
                        output_argument=out_func[out_rule_data[out_sum]][number];
                        out_val(out_rule_data[out_sum++],out_func,grade,
                                out_port[output_argument][0],out);
                }
        }
        for(i=0;i<1;i++){
                Y=XY=0.0;
                for(j=0;j<out_port[i][0];j++){
                        XY+=((j+1)*out[i][j]);
                        Y+=out[i][j];
                }
                if(Y==0){
                        printf("there is no rule to fit with the input data.¥n");
                        printf("please check the rules!¥n");
                        outportb(out_port[0][1],128);
                        exit(1);
                }
                output=(int)(XY/Y-1.0);
                outportb(out_port[i][1],output);
        }
}
outportb(out_port[0][1],128);
}
```

図6 Fuzzy to Cトランスレータが出力した倒立振子のCプログラム