**Introducere**

Logica Fuzzy  a fost definita in 1965 de catre prof. Lotfi Zadeh, de la Universitatea Berkeley. Spre deosebire de logica clasica, care lucreaza cu doua valori numerice exacte (0 pentru fals si 1 pentru adevarat), logica fuzzy foloseste o plaja continua de valori logice cuprinse in intervalul 0 - 1, unde 0 indica falsitatea completa, iar 1 indica adevarul complet. Astfel, daca in logica clasica un obiect poate apartine (1) sau nu (0) unei multimi date, in logica fuzzy putem defini gradul de apartenenta al obiectului la multime si care poate lua valori intre 0 si 1.

Logica fuzzy ofera instrumentele necesare pentru reprezentarea in sistemele inteligente a unor concepte imprecise cum sunt „mare”, „mic”, „scump”, „ieftin” s.a., concepte numite variabile lingvistice sau variabile fuzzy. Pentru reprezentarea acestora se folosesc seturile fuzzy, care capteaza din punct de vedere cantitativ interpretarea calitativa a termenilor. Bazate pe logica fuzzy, sistemele fuzzy sunt considerate un caz particular al sistemelor expert (motiv pentru care mai sunt denumite si sisteme expert fuzzy) care ofera o metoda flexibila pentru tratarea incertitudinii.

Calculul fuzzy se caracterizează prin faptul că permite manipularea conceptelor vagi care nu pot fi modelate prin concepte matematice exacte (numere, mulţimi sau funcţii clasice). Probleme în care intervin concepte vagi apar în teoria controlului când sistemele au caracter neliniar iar stările lor nu pot fi descrise în mod exact, ci doar prin enunţuri care au un grad de ambiguitate.

**Referinte Stiintifice**

In teza de masteratnumita ”**Fuzzy Logic Control for Aircraft Longitudinal Motion”,** a studentului ***Kashyapa Narenathreyas*** indrumat de ***Dr. Petr Hušek***se explica un control mai amanuntit al axei longitudinale ale aunui avion.

Acestia compara rezultatele obtinute pentru doua tipuri de control fuzzy: Sugeno si Mandami.

In primul rand acesta explica faptul ca design-ul aeronautic este compus din mai multi pasi precum design aerodinamic, design pentru controlul zborului, analiza structurii unui avion etc. Un element extrem de important in industria aeronautica este reprezentat de controul zborului avioanelor moderne, acestea bazandu-se in mare masura pe sisteme de control automate pentru majoritatea functiilor acestora, totodata cerintele unor sisteme de control cat mai avansate si mai fiabile fiind foarte mari.

Desi sunt dezvoltate multe tehnici de control a unui sistem dinamic, precum control PID (Proportional-Integral-Derivat), control LQ si control MPC, acestea nu sunt implementate pe aeronave datorita caracterului lor non-intuitiv. In industria aeronautica, unde siguranta este extrem de importanta, aceste tehnici de control nu ofera incredere suficienta pentru a fi instalate. In aceasta situatie, un sistem cu logica Fuzzy ar putea fi implementat, deoarece, asa cum am mentionat, permite manipularea conceptelor vagi, este neliniar, bazat pe un set de reguli si depinde de experienta umana si de intuitie.

S-a tinut cont de faptul ca in logica longitudinala a aeronavei actioneaza 3 forte importante: momentul de tangaj, rezistenta la inaintare si portanta.

Controlul tangajului preia feedback-ul de la unghiul de tangaj si produce o marime de intrare pentru eleron.

Pentru realizarea controlului de tip Mandami folosit un sistem de control fuzzy Mandami impreuna cu un sistem PI. In functie de aerodinamica aeronavei se primeste o variabila de intrare in sistemul fuzzy, iar iesirea acestui sistem reprezina intrarea in controller-ul PI.

Diagrama bloc care demonstreaza controlul este urmatoarea:



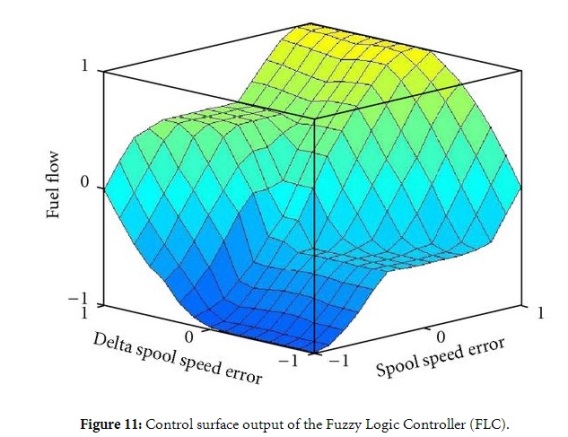
Figure 3-5: Hybrid Fuzzy Logic PI autopilot controllers for longitudinal system

Controlul fuzzy a fost realizat in Matlab folosind interfata sistemului fuzzy. Intrarile sistemului fuzzy sunt reprezentate de ungiul de tangaj si momentul de tangaj iar variabila de iesire este unghiul profundorului.

Functile de apartenenta folosite in lucrare sunt functii simple triunghiulare, cu diferite valori ale unghiurilor de intrare si iesire.

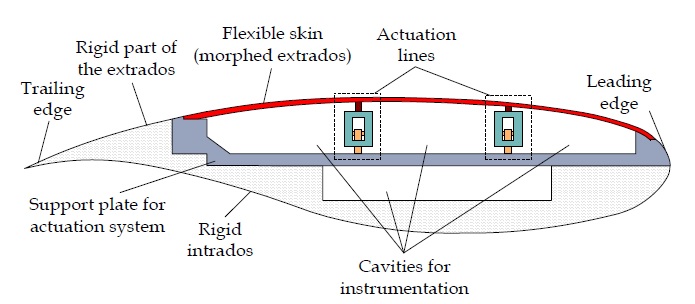
Concluziile acestul studiu au fost ca un sistem fuzii de tip Takagi-Sugeno este mai eficient decat unul de tip Mandami.

In lucrarea ”**Gas Turbine Engine Control Design Using Fuzzy Logic and Neural Networks**” autorii **[M. Bazazzadeh](http://www.hindawi.com/72814142/)**, [**H. Badihi**](http://www.hindawi.com/91684237/), si [**A. Shahriari**](http://www.hindawi.com/14743930/), se prezintă o abordare de succes în proiectarea unui controler Logic Fuzzy ( FLC ) pentru un anumit motor turbopropulsor . La început , un model matematic adecvat pentru motorul turbopropulsor este prezentat de ajutorul SIMULINK . Apoi, prin aplicarea de diferite funcții rezonabile pentru debitul de combustibil , se obțin o serie de parametri de funcționare ai motorului cum ar fi forța de tracțiune, creșterea limitei de compresie, temperatura la intrare in turbină , etc. Acești parametri sunt folositi ca o baza de date pentru a antrena o rețea neuronală . În a doua etapă , în funcție de această bază de date disponibila , se determina un număr diferit de functii pentru consumul de combustibil in functie de diverse regimuri de functionare ale motorului . Aceste funcții sunt folosite pentru a defini funcțiile fuzzy dorite, in functie de combustibil . Rețelele neuronale sunt folosite ca o metodă eficientă de a defini functiile optime de combustibil pentru modelul fuzzy . In ultima etapa se propune un controler fuzzy prin utilizarea modelului de simulare a motorului și a rezultatelor rețelei neuronale. Rezultatele din simularea modelului de motor cu logica fuzzy ilustrează că controlerul propus atinge performanța și stabilitatea dorită.



O alta intrebuintare a sistemelor fuzzy in domendiul aviatiei este prezentata de catre **Teodor Lucian Grigorie** si **Ruxandra Mihaela Botez**, din cadrul ***École de Technologie Supérieur – Canada,*** in lucrarea ”**New Applications of Fuzzy Logic Methodologies in Aerospace Field” .**

In aceasta lucrare autorii prezinta un nou concept de mecanism morphing, folosind materiale inteligente precum aliaj care memoreaza forma, ca elemente de actionare si tehnici de control fuzzy. Ei propun folosirea pe partea superioara a aripii, denumita si extrados, o pelicula subtire care permite deformarea profilului aripii astfel incat puctul de tranzitie a curentului de aer din laminar in turbulent sa se realizeze cat mai aproape de bordul de fuga a aripii.

  
Fig. 1. General architecture of the mechanical model

Scopul principal al lucrarii este acela de a realiza o reducere fortei de rezistenta, conditionata de schimbarea profilului aripii. Detectia punctului de tranzitie se face in functie de presiunea de pe suprafata aripii, masurata cu ajutorul senzorilor de presiune.

Pentru inceput s-au determinat profilele aerodinamice optime in functie de 35 de curgeri diferite ale aerului la 5 viteze diferite si 7 valori ale unghiului de atac. Profilele aerodinamice optime au fost folosite ca punct de incepere pentru design-ul sistemului de actionare a elementelor deformatoare ale profilului, acest sistem fiind controlat prin intermediul unui sistem fuzzy.

**Descrierea proiectului**

In cadrul acestui proiect vom incerca realizarea controlului unui profundor, care reprezinta elementul de control al inclinarii unei aeronave, folosimd sisteme cu logica de control Fuzzy.

Controlul automat al profundorului trebuie sa aiba loc doar in situatii de urgenta, cand se detecteaza o posibila pierdere a portantei. Detectia se va realiza in functie de inclinarea aeronvaei fata de axa longitudinala si de turatia motorului. O situatie in care aeronava se afla pe o traiectori ascendenta, deci o inclinare pozitiva, iar turatia motorului este scazuta duce la angajarea aeronvaei, adica la pierderea portantei, datorita curgerii turbulente a aerului pe suprafata aripii, in loc de o curgere laminara asa cum este normal. Pentru a preveni acest lucrul profundorul trebuie sa actioneze in mod automat si rapid, in sensul redresarii aeronvaei pentru a se reveni la o curgere laminara a aerului.



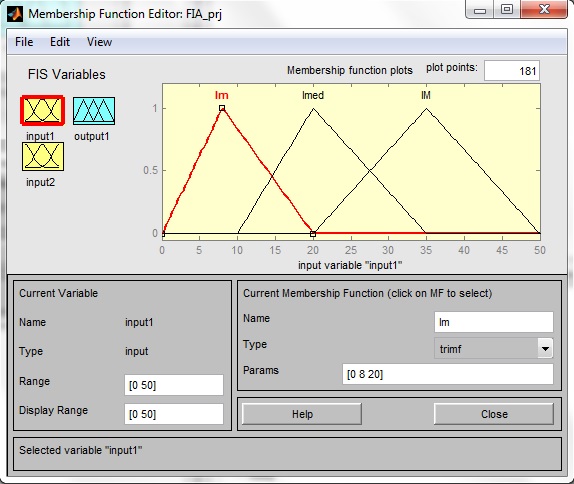
Axa de interes va fi axa longitudinala, datorita faptului ca miscarea profundorului duce la modificarea unghiului fata de aceasta axa.



Consideram ca variabile de intrare, inclinarea avionului si turatia motorului. Inclinarea avionului va fi determinata preluand informatiile de la un giroscop, iar turatia motorului va fi preluat tot de la instrumentele de bord ale avionului . Marimea de iesire este reprezentata de modificarea unghiului profundorului.

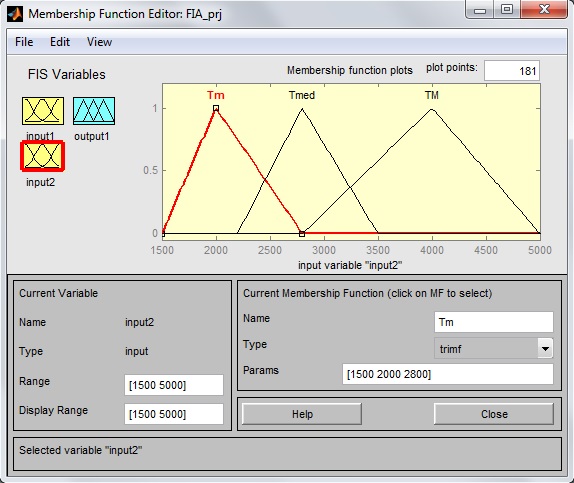
Inclinarea avionului este exprimata in grade si are urmatoarele valori:

* Inclinare mica : 0-20 grade;
* Inclinare medie: 10 – 35 grade;
* Inclianre mare : 20 – 50 grade.



Turatia motorului este exprimata in rotatii pe minut si are urmatoarele valori:

* Turatie mica:1500-2800 rpm;
* Turatie medie: 2200 -3500;
* Turatie mare: 2800 – 5000 rpm;

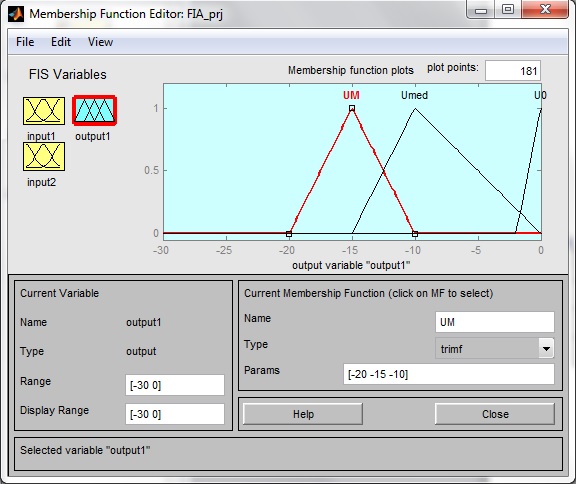


Marimea de iesire reprezentata prin ungiul profundorului va avea doar valori negative, acest lucru fiind necesar pentru a redresa avionul in cazul in care acesta este pe o panta ascendenta.

Gama de variatie a profundorului este cuprinsa intre urmatoarele valori:

* Unghi zero 0: -2 grade;
* Unghi mediu: 0: -15 grade;
* Unghi mare: -10: -20 grade;

Folosirea unghiului zero se explica prin faptul ca in anumite situatii sistemul nu trebuie sa interavina asupra profundorului, la decolare spre exemplu unde aeronava are o inclintie pozitiva, asta insemnand ca sistemul nu trebuie sa intre in functiune in orice moment se detecteaza o inclinare pozitiva a aeronavei, ci doar atunci cand exista riscul pierderii portantei.



In functie de aceste valori ale variabilelor de intrare si de iesire vom realiza tabela de reguli pentru controlul unhgiului profundorului. Avem in vedere faptul ca acest sistem de reguli are rolul de a preveni pierderea portantei si nu de a impiedica aeronava sa castige inaltime.

Tabela de reguli este urmatoarea:

|  |  |  |  |
| --- | --- | --- | --- |
| Inclinare \ Turatie | Turatie mica (Tm) | Turatie medie (Tmed) | Turatie mare (TM) |
| Inclinare mica (Im) | Unghi med | Unghi zero | Unghi zero |
| Inclianre medie (Imed) | Unghi med | Unghi zero | Unghi zero |
| Inclinare mare (IM) | Unghi mare | Unghi med | Unghi zero |

In urma acestor considerent rezulta urmatoarele reguli:

IF Inclinare mica AND Turatie mica THEN Unghi med;

IF Inclinare mica AND Turatie medie THEN Unghi zero;

IF Inclinare mica AND Turatie mare THEN Unghi zero;

IF Inclinare medie AND Turatie mica THEN Unghi med;

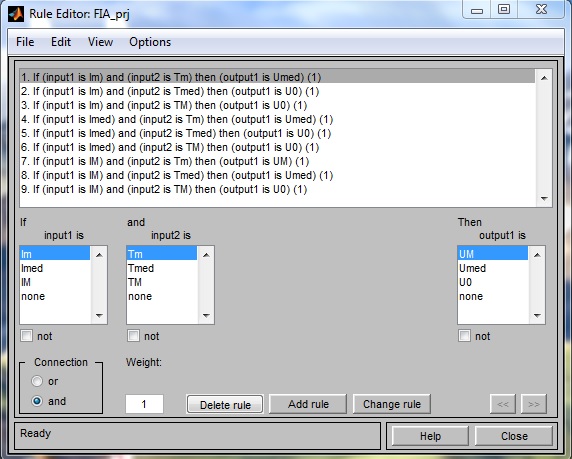
IF Inclinare medie AND Turatie medie THEN Unghi zero;

IF Inclinare medie AND Turatie mare THEN Unghi zero;

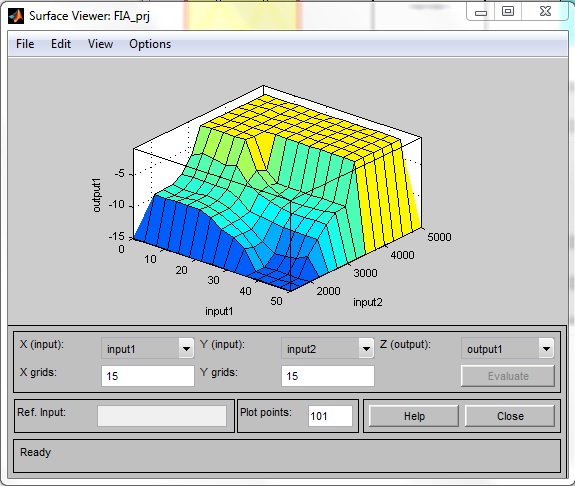
IF Inclinare mare AND Turatie mica THEN Unghi mare;

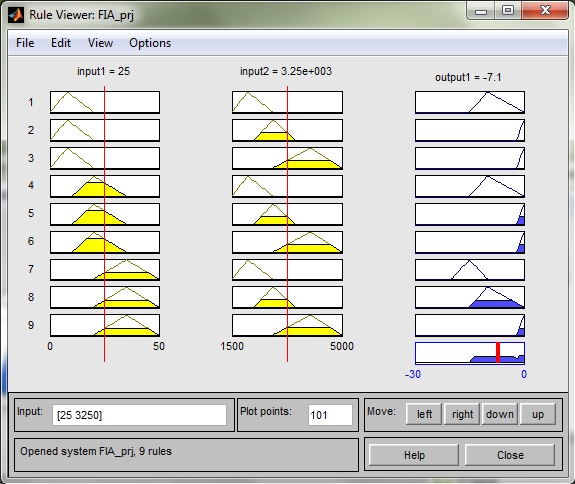
IF Inclinare mare AND Turatie medie THEN Unghi mediu;

IF Inclinare mare AND Turatie mare THEN Unghi zero;



Graficul regulilor si suprafata rezultate in urma simularii in programul Matlab sunt urmatoarele:





**Algoritmul sistemului fuzzy**

#include <stdio.h>

#include <conio.h>

#include <math.h>

#define min(a,b) ((a<b)?a:b)

#define max(a,b) ((a<b)?b:a)

#define nmfx1 3

#define nmfx2 3

#define nmfy 3

#define npcteR 100

Definirea inrarilor si a functiilor de apartenenta

float x1[nmfx1][3]= {{0,8,20},{10, 20,35},{20,35,50}};

float miux1[nmfx1][3]= {{0,1,0},{0, 1,0},{0, 1,0}};

int npx1 =3;

float x2[nmfx2][4]= { {1500, 2000, 2800},{2200, 2800, 3500},{2800, 4000, 5000}};

float miux2[nmfx2][4]= {{0,1,0},{0,1,0},{0,1,0}};

int npx2 =3;

Definirea iesirii, a functiilor de apartenteta a iesirii

float y[nmfy][3]= {{-20, -15, -10},{-15, -10, 0},{-2, 0, 0}};

float miuy[nmfy][3]= {{0,1,0},{0, 1,0},{0,1,0}};

int npy =3;

Definirea tabelei de reguli

int reg [nmfx1][nmfx2]= { {1,2,2},{1,2,2},{0,1,2}};;

float grad\_apartenenta (float, float\*,float\*, int);

int elimin\_pct\_colin(float \*, float\*, int, float \*, float \*);

float cog(float \*, float \*, int);

void main(void)

{

float x1c, x2c, gax1[nmfx1], gax2[nmfx2], gay[nmfy], m;

float R[npcteR+1], miuR[npcteR+1], nR[22], nmiuR[22], yinf=-20, ysup=0, step, yc; // yinf = y[0][0], ysup = y[nmfy] [3]

int i, j, np;

printf ("x1="); scanf ("%f",&x1c);

printf ("x2="); scanf ("%f",&x2c);

for (i=0; i<nmfx1; i++)

{

gax1[i]=grad\_apartenenta(x1c, x1[i], miux1[i], npx1);

}

for (i=0; i<nmfx2; i++)

gax2[i]=grad\_apartenenta(x2c, x2[i], miux2[i], npx2);

for (i=0; i<nmfy; i++)

gay[i] =0;

Reguli Fuzzy

for (i=0; i<nmfx1; i++)

for (j=0; j<nmfx2; j++)

if(gax1[i]!=0 && gax2[j] != 0)

{

m=min (gax1[i],gax2[j]);

if(m>gay[reg[i][j]])

gay[reg[i][j]]=m;

}

//reuniune

step = (ysup - yinf)/npcteR;

for (i=0, yc=yinf; i<=npcteR ; i++, yc+=step)

{

m=0;

for(j=0; j<nmfy; j++)

{

m=max(m, min(gay[j], grad\_apartenenta(yc, y[j],miuy[j], npy)));

}

miuR[i]=m;

R[i]=yc;

// printf("(%.1f %.2f)", R[i], miuR[i]);

}

Secventa de defuzificare

np=elimin\_pct\_colin(R, miuR, npcteR+1, nR, nmiuR);

yc= cog(nR, nmiuR, np);

printf("Valoare iesire este=%.2f\n",yc);

getch();

}

float grad\_apartenenta(float xc, float x[], float miux[], int npx)

{

if (xc<x[0]) return miux[0];

for (int i=0; i<npx-1;i++)

if (xc>=x[i]&&xc<=x[i+1])

{

// return (xc-min(x[i]+(miux[i]>miux[i+1]), x[i+1]))/(x[i+1]-x[i])\* (miux[i+1]-miux[i])+min (miux[i], miux[i+1]);

return (xc - x[i+(miux[i]>miux[i+1]) ]) / (x[i+1] - x[i])\* ( miux[i+1] - miux[i])+ min(miux[i],miux[i+1]);

}

if (xc>x[npx-1]) return miux[npx-1];

}

float cog(float y[], float miuy[], int npy)

{

float s1, s2, arie, cg, arie2, cg2;

s1=s2=0;

for(int i=0; i<npy-1; i++)

{

if(miuy[i]==miuy[i+1]) //dreptunghi

{

arie=miuy[i]\*(y[i+1]-y[i]);

cg= (y[i+1]+y[i])/2;

}

else if (miuy[i]==0 || miuy[i+1]==0) //triunghi

// if (!miuy[i]\*miuy[i+1])

{

// arie=(y[i+1]-y[i])\* (float)fabs((double)(miuy[i]-miuy[i+1]))/2;

arie=(y[i+1]-y[i])\*(miuy[i]+miuy[i+1])/2;

cg=(y[i]+y[i+1]+ (miuy[i]>miuy[i+1]?y[i]:y[i+1]))/3;

// cg=y[i]\*(1+miuy[i]>miuy) +y[i+1]\*(1+miuy[i]<miuy)/3;

}

else //trapez

{

arie=min(miuy[i],miuy[i+1])\*(y[i+1]-y[i]);

cg= (y[i+1]+y[i])/2;

// arie2=(y[i+1]-y[i])\*(float)fabs((double)(miuy[i]-miuy[i+1]))/2;

arie2=(y[i+1]-y[i]) \* (max(miuy[i], miuy[i+1])-min(miuy[i], miuy[i+1]))/2;

//cg=(y[i]+y[i+1]+ (miuy[i]>miuy[i+1]?y[i]:y[i+1]))/3;

cg2=(y[i]+y[i+1]+ (miuy[i]>miuy[i+1]?y[i]:y[i+1]))/3;

cg=(arie\*cg+arie2\*cg2)/(arie+arie2);

arie+=arie2;

}

// printf("Arie%d= %f, cg%d= %f\n", i+1, arie, i+1, cg);

s1+=arie\*cg;

s2+=arie;

}

return s1/s2;

}

int elimin\_pct\_colin(float R[], float miuR[], int p, float nR[], float nmiuR[])

{

int np, i;

float x1, x2, x3, y1, y2, y3, m1, m2; //m1, m2 reprez pantele

x1=R[0]; y1=miuR[0]; x2=R[1]; y2=miuR[1]; x3=R[2]; y3=miuR[2];

np=0;

for(i=3; i<p; i++)

{

m1=(y2-y1)/(x2-x1); m2=(y3-y2)/(x3-x2);

if(fabs ((y2-y1)/(x2-x1)- (y3-y2)/(x3-x2))> 0.0001)

{

nR[np] =x1; nmiuR[np] =y1; np++;

x1=x2; y1=y2;

}

x2=x3; y2=y3;

x3=R[i]; y3=miuR[i];

}

nR[np] =x1; nmiuR[np] =y1; np++;

// m1=(y2-y1)/(x2-x1); m2=(y3-y2)/(x3-x2);

if (fabs ((y2-y1)/(x2-x1)- (y3-y2)/(x3-x2))>0.0001)

{

nR[np] =x2; nmiuR[np] =y2; np++;

}

nR[np] =x3; nmiuR[np] =y3; np++;

for(i=0; i<np; i++)

printf("(%.1f %.2f)\n ", nR[i], nmiuR[i]);

return np;

Rezultatele obtinute folosind programul scris in C si simularea in MatLab:

|  |  |  |  |
| --- | --- | --- | --- |
| X1(inclinare) | X2 (turatie) | Yc (unghi) – C | Yc (unghi)-Matlab |
| 35 | 2000 | -15 | -15 |
| 20 | 2800 | -0,73 | -0,57 |
| 8 | 2000 | -8,33 | -8,33 |
| 30 | 2000 | -11,42 | -11,4 |
| 35 | 3000 | -8,09 | -8,05 |
| 8 | 4000 | -0,73 | -0,57 |

Comparand rezultatele obtinute se observa ca acestea au valori apropiate, acest lucru demonstrand corectitudinea programului. Cu toate acestea valoarea maxima a inclinarii profundorului este limitata la -15 grade. Modificarea functiilor de apartententa ar putea mari aceasta limita.

**Studii de caz**

Un prim studiu de caz consta in modificarea tabelei de reguli si observarea modificarilor ce apar in functionarea sistemului. Vom considera urmatoarea tabela de reguli:

|  |  |  |  |
| --- | --- | --- | --- |
| Inclinare \ Turatie | Turatie mica (Tm) | Turatie medie (Tmed) | Turatie mare (TM) |
| Inclinare mica (Im) | Unghi zero | Unghi zero | Unghi zero |
| Inclianre medie (Imed) | Unghi mare | Unghi mediu | Unghi zero |
| Inclinare mare (IM) | Unghi mare | Unghi mare | Unghi zero |

Rezultatele obtinute in urma simularii sunt urmatoarele:

|  |  |  |  |
| --- | --- | --- | --- |
| X1(inclinare) | X2 (turatie) | Yc (unghi)-C | Yc(unghi)-MatLab |
| 8 | 1700 | -0,87 | -0,74 |
| 5 | 4900 | -0,9 | -0,1 |
| 45 | 2800 | -15 | -15 |
| 30 | 2800 | -11,4 | -11,4 |
| 30 | 4500 | -0,83 | -0,74 |

Aceste valori demonstreaza faptul ca sistemul este stabil si poate functiona bine si in cazul schimbarii tabelei de reguli asemenea celei de mai sus. Totusi, schimbarea tabelei de reguli a fost gandita astfel incat sa permia aeronavei castigul de inaltime. O schimbare majora a tabelei de reguli precum :IF Inclinare mica AND Turatie mare THEN Unghi mare ar fi dus la incapacitatea aeronavei de putea castiga inaltime, sistemul fiind in acest caz daunator.

Al doilea studiu de caz presupune modificarea functiilor de apartenenta pentru a mari valoarea limita a inclinarii profundorului. In acest sens am modificat functiile de apartenenta ale variabilei de iesire cu urmatoarele valori :

* Unghi 0: -2 grade;
* Unghi mediu: 0: -17 grade;
* Ungi mare : -15: -25 grade.

Aceste modificari le-am implementat si in algoritmul scris in C, iar mai apoi inMatlab.

Pentru verificare am considerat inclinarea aeronavei de 40 de grade si turatia motorului de 2000 rpm, tabela de reguli fiind aceasi folosita si in proiec. La iesire am obtinut valoarea de -18.19 grade iar folosind simularea Matlab am obtinut -20 de grade. S-a dovedit faptul ca prin modificarea functiilor de apartenenta de la iesirea sistemului se poate modifica valoarea unghiului profundorului, lucru necesar in realizarea unui sistem care sa poatea fi implementat in functie de diferite cerinte.