SLIDING MODE CONTROL WITH FIXED AND FUZZY SLIDING SURFACES

A DISSERTATION

Submitted in partial fulfilment of the requirements for the award of the degree

of

MASTER OF ENGINEERING

in

ELECTRONICS AND COMMUNICATION ENGINEERING

(With Specialization in Control and Guidance)

By

RAKESH KUMAR ARYA

DEPARTMENT OF ELECTRONICS AND COMPUTER ENGINEERING

UNIVERSITY OF ROORKEE

ROORKEE–247 667 (INDIA)

FEBRUARY, 2000

CANDIDATE'S DECLARATION

I hereby declare that the work which is being presented in the dissertation entitled **"SLIDING MODE CONTROL WITH FIXED AND FUZZY SLIDING SURFACES"** in partial fulfilment of the requirements for the award of the Degree of **Master of Engineering (M.E.) in Control and Guidance** in the Department of Electronics and Computer Engineering, University of Roorkee, Roorkee, is an authentic record of my own work carried out by me from October, 1999 to January, 2000 under the supervision and guidance of **Dr. R. Mitra**, Professor, E & C Department, University of Roorkee, Roorkee, U.P., India.

The matter embodied in this dissertation has not been submitted by me for the award of any other degree or diploma.

Dated : 8 February, 2000
Place : Roorkee

(RAKESH KUMAR ARYA)

This is to certify that the above statement made by the candidate is correct to the best of my knowledge and belief.

(Dr. R. Mitra)
Professor
E & C Department
University of Roorkee
Roorkee-247 667, U.P.,
INDIA

ACKNOWLEDGEMENT

It gives me great pleasures to take this opportunity to thank and express my deep sense of gratitude to my guide **Dr. R.Mitra**, Professor, E&C Department, University of Roorkee, for his valuable suggestions, guidance and constant envouragement during the course of this dissertation work. I deem it my privilege to have carried out this dissertation under his able guidance.

I would like to thank all my friends and well wishes who helped me directly or indirectly in making this report.

I am also thankful to my wife, she always encourages me to do my working taking hard pains during my M.E.

Finally, I express my regards to my father and mother, who has been a constant source of inspiration to me.

(RAKESH KUMAR ARYA)

ABSTRACT

Reaching time plays an important role for the multi-coupled system like robot manipulator to achieved the minimum reaching time various method has been suggested. Variable structure system with sliding mode is one of the technique. Fuzzy controlled switching surface minimizes the reaching time up to distinct level.

Instead of using conventional sliding mode controller having linear time varying switching surface, if fuzzy logic is used to regulate the switching surface, the reaching time of the system trajectory is shorter than in the fixed method and having better performance than the conventional method in reaching time.

In this dissertation the fuzzy controller is used to regulate the switching surface and its performance is compared with fixed switching surface. The proposed FLC is designed using a very simple control rule base and most natural and unbiased membership function (symmetrical triangles with equal bases and 50%, overlap with neighboring MFS)

LIST OF SYMBOLS AND ABBREVIATIONS

Symbol	Meaning
$\in$	belong to
$\notin$	does not belong to
$\forall$	for all
$\Rightarrow$	equivalent to
$\vert$	such that
max	maximum
min	minimum
$\dot{x}$	derivative of x
x_d	desired value of x
e	error
sgn	sign function
λ	lemda
μ_A	membership function
η	eta
δ	delta

CONTENTS

CHAPTER - 1

INTRODUCTION

1.1 INTRODUCTION

Conventional model based control has the advantage that one can prove optimality and stability; however, there are difficulties in dealing with non-linear, dynamic and ill-understood processes which are common in real world.

Variables structure system (VSC) [1,3,6] with sliding mode has different structure on both sides of sliding surface is the major practical technology that is widely used in the control of Robotic manipulator [10], large scale control system, general non-linear control system [5], robust motion control, flexible structure control of space craft. It has been recognized as a powerful design technique suitable for complex and non-linear system with complicated interaction and uncertainties.

Sliding mode (SM) controller with fixed sliding surface has the following characteristics (i) Robust stability (ii) Linearisation (iii) Order reduction. However it become sensitive to parameter uncertainties and noise disturbances in the reaching phase. The design of sliding surface is therefore one of the factor in the system performance.

To eliminate the system sensitivity in reaching phase, various methods have been suggested, (i) High gain feedback was used to minimize the reaching phase, unfortunately this may cause sensitive to unmodelled dynamics and chattering which is undesirable in physical system, (ii) A time varying sliding surface was proposed to remove the reaching phase by imposing a constrains that initial error be zero in tracking control and (iii) The system trajectory moved on the sliding surface from arbitrary initial points, facing problem that the reaching time to an equilibrium point increase and sensitivity against disturbances.

Sliding mode controller with fuzzy sliding surface was proposed and tested for robotic manipulator by Takagi T. and Sugeno M. in 1998 [6].

Fuzzy logic controller (FLC's) have been suggested as a promising alternative approach for designing the sliding surface, especially those that are two complex for analysis by conventional technique. The effective control strategies that the human operator learn through his experience or by using common sense can often be expressed as a set of condition-action rules (called fuzzy rules), which describe condition about the process state using linguistic terms (i.e. fuzzy sets such as *low, medium, high, slightly positive*) and recommended control action using linguistic terms such as *increase* slightly or *decrease moderately*.

An example of such a rule is the given below :

IF error is *small negative* AND change of error is *big positive* or *medium positive* THEN decrease the stream flow *slightly*.

Since E.H. Mamdani introduced the concept of fuzzy logic control in 1974, which was strongly motivated by theory of fuzzy sets developed by L.N. Zadeh [2]. Takagi. T and Sugeno M. have done the various works in the field of fuzzy sliding mode controller and identification of systems. FLC-based systems have proven to be superior in performance to conventional systems in areas such as process control, automatic train operating systems, artificial intelligence, advances in computer hardware technology supporting fuzzy control have resulted in numerous commercial FLC applications such as washing machines, *Vacuum* cleaners, *air conditioner* etc.

Compared to conventional technique, FLC offers three important benefits, first, developing a FLC is cheaper than developing a model based on other controller with equivalent performances, second FLC's are more robust then conventional SM controller because they can improve the performance. Third, FLC's are customizable, since it is easier to understand and modify their rule.

A major limitation of fuzzy control is the lack of a systematic methodology for developing fuzzy rules. A set of fuzzy rules often needs to the manually adjusted on trial and error basis before it reaches the desired level of performance.

Multi-coupled system like robot manipulator where the output of one is linked with the input to other the reaching time is important, to reduce the reaching time the structure of the system is varied called VSC and, if the systems's trajectory always pointed toward the sliding surface then the system has sliding mode.

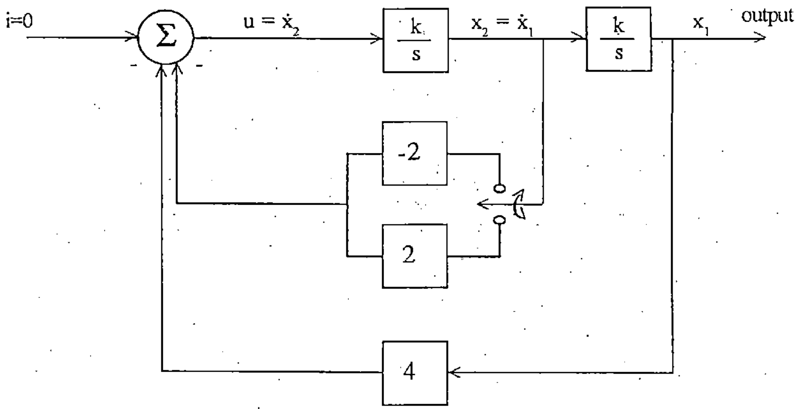


Fig.1.1 : Schematic diagram of controlled process

In this dissertation the sliding mode controller is simulated with fuzzy sliding surface, which improved the chattering and decrease the reaching time of the system.

1.2 STATEMENT OF PROBLEM

The process chosen to investigate is a double integrator plant with variable structure as shown in fig.1.1. The structure of the system is change such that the moving representative point of the system is constrained to move along a predetermined switching surface.

The state space representation of the system is given by

$$\dot{x}_1 = kx_2$$

$$\dot{x}_2 = ku$$

$$u = = -2\text{sgn}(x_2)\, x_2 - 4x_1$$

$$\text{sgn}(x_2) \begin{cases} = +1 \text{ when } S(x_1, x_2) > 0 \\ = -1 \text{ when } S(x_1, x_2) < 0 \end{cases}$$

The time varying sliding surface is defined by

$$S(x_1, x_2) = x_1 + \lambda x_2$$

The slope λ of the switching line is regulated by the fuzzy controller to obtain the fuzzy switching surface.

The problem addressed in this dissertation is to compare the performance (reaching time) of the SM controller with fixed and fuzzy sliding surface. The gain of the system is varied and performance is compared.

1.3 ORGANIZATION OF DISSERTATION

Including this introductory chapter, which gives a brief description about the fuzzy control and its application, the dissertation is organised as follows :

In the second chapter important terms and definition about the fuzzy set theory are discussed, which are the back bone of fuzzy control.

The third chapter describes the general structure of fuzzy controller. Design parameters of different module of fuzzy control have been discussed in this chapter.

Chapter 4 discusses the design consideration used in the dissertation for fuzzy controller, process and SM controller. It also discusses the detail of software.

Chapter 5 discusses the simulation results obtained and compares the responses for fuzzy and SM controllers.

Chapter 6 concludes the dissertation.

1.4 LITERATURE REVIEW

Variable structure system (VSC) theory was developed during last 30 years exclusively in USSR. Utkin (1977, 78) developed the theory of VSC with sliding mode.

The basic mathematical ideas of non-linear system with discontinuous right hand side comes from theory developed by Fillipov (1960). In SMC, the moving representative point of the system is constrained to move along a predetermined switching surface Itkin (1976). The design of switching surfaces completely determine the performance of the system. The robustness of VSC can be improved by shortening the time required to attain the sliding mode.

Young, et al. (1977) used the high gain feedback to speed up the reaching phase. This may cause sensitive to unmodelled dynamics and chattering. Slotine J.J. and Sastry, S.S. (1983) suggested a time varying switching surface, facing problems that reaching time to an equilibrium point increases and also sensitivity against disturbance increases.

Sliding mode controller with fuzzy sliding surface is designed by Takagi T. and Sugeno M. (1998), which has better performance than conventional SM controller.

CHAPTER - 2

FUZZY SETS : MATHEMATICS OF FUZZY CONTROL

2.1 INTRODUCTION

Fuzzy set theory was developed in 1965 by Lofti Zadeh of the University of California in Berkeley [2]. This approach is useful to solve the typically complex problem which are after left to deal with human being.

Fuzzy set theory is based on ordinary set theory (classical set theory) and becomes identical with it in the limiting case where the properties being dealt with the 'crisp'. As with ordinary sets, fuzzy set are defined over some universe discourse, which might be a population of people, a set of possible measurement values, a range of possible output voltage, or otherwise depending on the problem.

2.2 FUZZY SETS

In fuzzy set theory, 'normal' set is called crisp set, in order to distinguish them from fuzzy set let C be a *crisp sets* defined on the universe U, then for any element u of U, either $u \in C$ or $u \notin C$. In fuzzy set theory this property is generalized, therefore in fuzzy set F, it is not necessary that either $u \in F$ or $u \notin F$.

The *characteristics function* of a crisp set assign a value of either 1 or 0 to each individual in the *universal set*, thereby discriminating between *members* and *nonmembers* of the crisp set under consideration. This function can be generalized such that the values assigns to the element of the universal set fall within a specified range and indicate the membership grade of these element in the set in question. Larger values denote higher degree of set membership. Such a function is called a *membership function*, and the set defined by it a *fuzzy set*.

The most commonly used of values of membership function is the unit interval [0, 1]. In this case each membership function maps element of a given universal set X, which is always a crisp set, into real member in [0, 1].

Two distinct notation are most commonly employed in the literature to denote membership function of a fuzzy set A is denoted by μ_A that is

$$\mu_A : X \to [0, 1]$$

In the other one the function is denoted by A and has form

$$A : X \to [0, 1]$$

Definition : The membership function μF of a fuzzy set F is a function

$$\mu_F : U \to [0, 1]$$

So, every element u from U has a membership degree $\mu_F(u) \in [0,1]$.F is completely determined by the set of tuples

$$F = \{(u, \mu_F(u)) \mid u \in U\}.$$

2.2.1 Properties of Fuzzy Sets

Let A and B be fuzzy sets defined respectively on the universe X and Y and let R be a fuzzy relation defined as X x Y.

(i) ***Support*** : The support of a fuzzy set A is the crisp set that contains all elements of A with non zero membership degree denoted by S(A) mathematically defined as

$$S(A) = \{u \in X \mid \mu_A(u) > 0\}$$

The support of fuzzy set is an interval

(ii) ***Width*** : The width of a convex fuzzy set A with support set S(A) is defined as;

$$\text{Width}(A) = \sup(S(A)) - \inf(S(A))$$

sup and int. denote the mathematical operations supremum and infimum. They are defined as,

$$\alpha = \sup(A) \text{ iff } \forall_x \in A : x \leq \alpha \text{ and } \forall \in > 0\ \exists_x \in A : x > \alpha - \in ,$$

$$\beta = \inf(A) \text{ iff } \forall_x \in A : x \geq \beta \text{ and } \forall \in > 0\ \exists_x\ A : x < \beta + \in$$

for the fuzzy set A with membership function $\Lambda(x : \alpha, \beta, \gamma)$. Its support set is $S(A)=[\alpha,\gamma]$ its width is width $(A)=\gamma-\alpha$ left width $(A)=\beta-\alpha$, right width $(A)=\gamma-\beta$.

(iii) ***Nucleus*** : The nucleus of a fuzzy set A is the crisp set that contains all values with membership degree formally, mathematically defined as

$$\text{nucleus }(A) = \{u \in X \mid \mu_A(u) = 1\}$$

If there is only one point with membership degree equal to 1, then this point is called the peak value of A

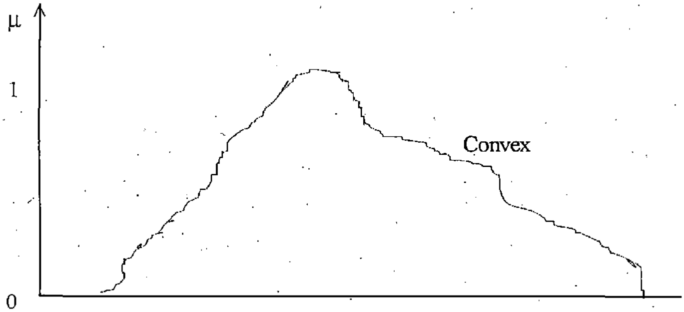
The nucleus is the interval [20,24], the supports is interval [18,26].

(iv) ***Height*** : The height of a fuzzy set A is equal to the largest membership degree μ_A mathematically

$$\text{hgt}(A) = \sup_{u \in X} \mu_A(u)$$

A fuzzy set is normal if $\text{hgt}(A)=1$ and subnormal if $\text{hgt}(A) < 1$.

(v) ***Convexity*** : A fuzzy set is convex if its membership function does not contain 'dips' this means that the membership function is for example, increasing, decreasing or bell shaped. Mathematically a fuzzy set is convex if and only if



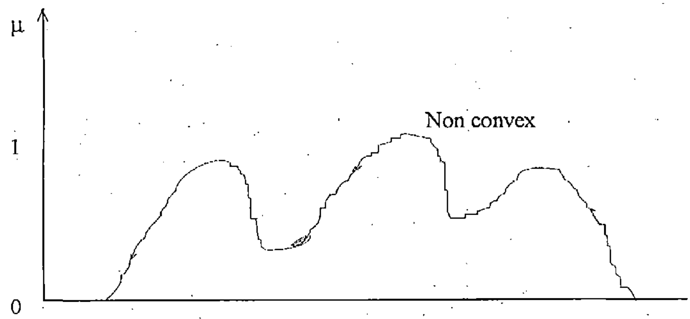


Fig. 2.1 : An example of convex and a non convex (cancave) fuzzy set

$$\forall x,y \in X\ \forall \lambda \in [0, 1] : \mu_A (\lambda.x + (1-\lambda).y) \geq \min (\mu_A (x), \mu_A(y))$$

Fig.2.1 show a convex and concave function.

2.2.2 Operations of Fuzzy Sets

(i) ***Equality :***

Two fuzzy sets are equal (A=B) if and only if

$$\forall_x \in S : \mu_A(x) = \mu_B(x)$$

(ii) ***Subset :***

A is a subset of B ($A \subseteq B$) if and only if

$$\forall\ x \in X : \mu_A(x) \leq \mu_B(x)$$

(iii) ***Union :***

Union of two fuzzy sets A and B is given by

$$\forall\ x \in X : \mu_{A \cup B}(x) = \max (\mu_A(x), \mu_B(x))$$

diagrammatic representation of union is given in fig.2.2.

(iv) ***Intersection :***

Intersection of two fuzzy sets A and B is given by

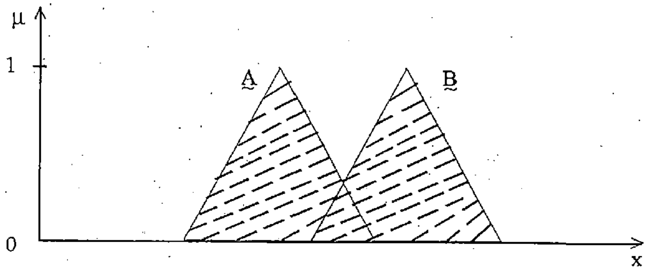


Fig. 2.2 : Union of fuzzy sets A and B

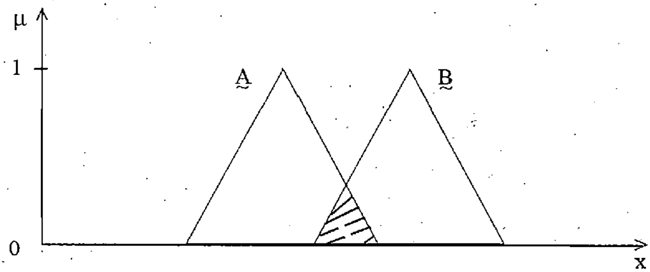


Fig. 2.3 : Intersection of fuzzy sets A and B

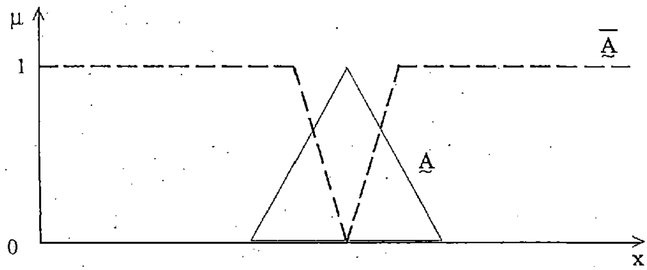


Fig. 2.4 : Complement of fuzzy set A

$\forall x \in X : \mu_{A \cap B}(x) \min (\mu_A(x), \mu_B(x))$

Diagrammatic representation of intersection is given in fig.2.3.

(v) *Complement:*

Complement of a fuzzy set A is given by

$\forall x \in X : \mu^{-}_{A}(x) = 1 - \mu_A(x)$

Diagrammatic representation of complement is given in fig.2.4

2.2.3 Fuzzy Proposition

Approximate reasoning is used to represent and reason with knowledge expressed in atomic primitives, which are expressed in a natural language form, example,

"Error has the value negative big"

The above natural language expression is rewritten as

"Error has the property of being negative big"

Symbolically it is written as

e is NB

where 'is' stands for "has the property of being".

The 'meaning' of the symbolic expression "E is NB" helps us decide the degree to which this symbolic expression is satisfied given a specific physical value of error.

Based on the notion of atomic fuzzy proposition and linguistic connective such as 'and ', 'or', 'not' and 'IF-THEN', one can form more fuzzy proposition called compound fuzzy proposition e.g.

X is A and X is B,

X is A or X is B,

X is not A

(X is A and X is not B) or X is C

if X is A than X is B, etc.

'and' = conjunction 'or' = disconjunction 'not' = negation

(i) Conjunction : 'and'

If A and B are two fuzzy set defined over universe of discourse U if p or q be the following two atomic fuzzy proposition P : "X is A" and q : "X is B" then conjunction (Λ) is defined as

Symbolic	**Meaning**
X is A,	μ_A or $\widetilde{A}$
X is B,	μ_B or $\widetilde{B}$
$\therefore$ X is $A \cap B$	$\therefore$ $\mu_{A \cap B}$ or $\widetilde{A} \cap \widetilde{B}$

(ii) ***Disconjunction ' or' :***

To disconjunction ($\cup$) is given by for the same fuzzy set and fuzzy preposition:

Symbolic	**Meaning**
X is A,	μ_A or $\widetilde{A}$
X is B,	μ_B or $\widetilde{B}$
$\therefore$ X is $A \cup B$	$\therefore\ \mu_{A \cup B}$ or $\widetilde{A} \cup \widetilde{B}$

(iii) ***Negation ' not' :***

The negation "X" is not A of a fuzzy preposition "X is A" is given by

Symbolic	**Meaning**
X is A,	μ_A or $\widetilde{A}$
$\therefore$ X is A'	$\therefore\ \mu_A'$ or $\widetilde{A}\,'$

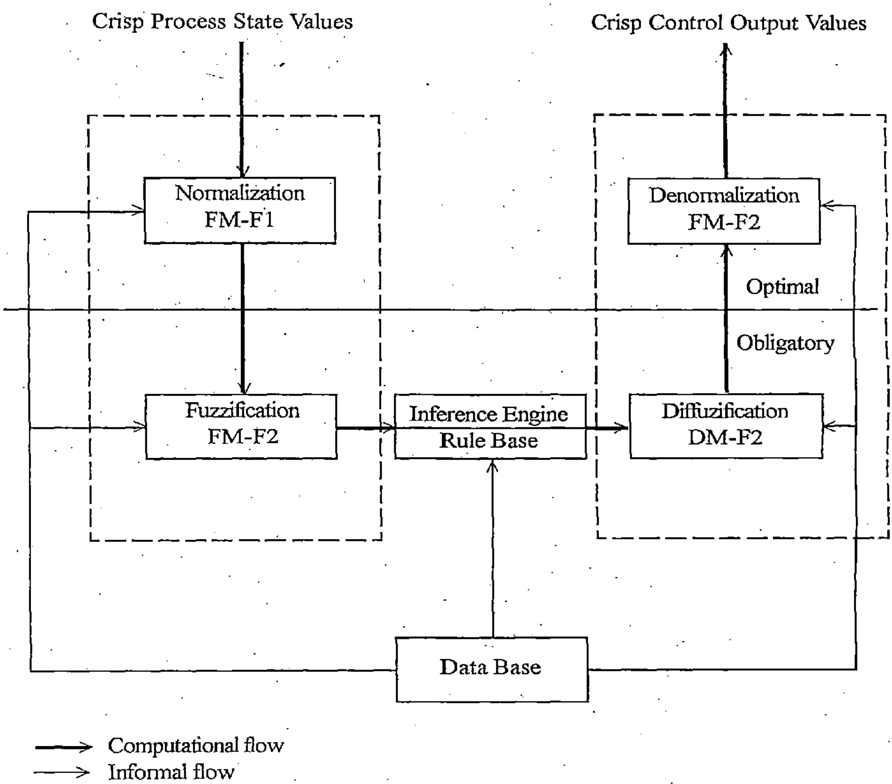


Fig.3.1 : The structure of FKBC

FUZZY CONTROLLER AN OVERVIEW

Fig.3.1 shows the general structure of fuzzy knowledge base controller (FKBC). As illustrated in figure. It consists of the following components,

- Fuzzification module
- Knowledge base
- Inference
- Defuzzification module

3.1 FUZZIFICATION MODULE

The fuzzification model (FM) performs the following functions [4]:

3.1.1 FM-F1

This module performs the scale transformation (i.e. input normalization) which maps the physical values of the current process state variables into a normalized universe of discourse (normalized domain). When a non-normalized domain is used then there is no need of FM-F1.

3.1.2 FM-F2

This module converts the normalized values from above to the fuzzy sets i.e. it convert a point wise (crisp), current values of a process state variable into a fuzzy set.

The choice of fuzzification strategy is determined by the type of the inference engine or rule firing.

3.2 KNOWLEDGE BASE

The basic function of the data base is to provide the necessary information for the proper functioning of the fuzzification module, the rule base, and the defuzzification module. This information includes [2] :

- Fuzzy sets representing the meaning of the linguistic values of the process state and control output variable.
- Physical domains and their normalized counterparts together with the normalization/denormalization (scaling) factors. The knowledge base of a FKBC consists of a data base and rule base.

3.2.1 Data Base

The design parameter of the data base includes :

- Choice of membership functions
- Choice of scaling factors.

For example, a PI like fuzzy controller can be expressed as

$$N_{\Delta u}\ \Delta u(k) = F\ (N_e\ e(k)),\ N_{\Delta e}\ .\ \Delta e(k))$$

Where

N_e, and $N_{\Delta e}$ are the scaling factors for e, Δe and Δu respectively.

The basic approaches for the determination of the scaling factor is heuristic and formal. The performance criteria are

- Desired value of overshoot
- Desired rise time
- Desired amplitude of oscillation

3.2.2 Rule Base

The design parameter of the rule base include

- Choice of process state and control output variables
- Choice of the content of the rule antecedent and the rule consequent
- Choice of term sets for the process state and control output variables.
- Derivation of the set of rules

(i) Choice of membership function

Let the physical domain of e, Δe, Δu, be ε, $\Delta\varepsilon$, Δu where e(error), Δe(change of error) are input variables and Δu(change in control) is output variable of fuzzy controller.

The meaning or interpretation of a particular linguistic value LX of linguistic variable x is given by a fuzzy set $\tilde{L}X$ or μ_{LX} defined on the domain (universe of discourse) x of x as

$$\tilde{L}X = \mu_{LX} = \int_x \mu_{LX}(x) / x$$

Now suppose that $\sigma E = \sigma\Delta E = \sigma\Delta U = \{NB, NM, NS, ZE, PS, PM, PB\}$ i.e. the term sets containing the linguistic values for the three linguistic variable are the same. In this case there is need to define twenty one membership function representing the meaning of each linguistic value from the above term set on the respective domain ε, $\Delta\varepsilon$ and Δu. For computational efficiency and efficient use of memory, a uniform representation of the membership functions is required. The uniform representation can be achieved by employing membership functions with uniform shape and parametric, functional definition.

The most popular choices for the shape of the membership function include triangular, trapezoidal and bell-shaped functions. These choices can be explained by the ease with which a parametric, functional description of the membership function can be

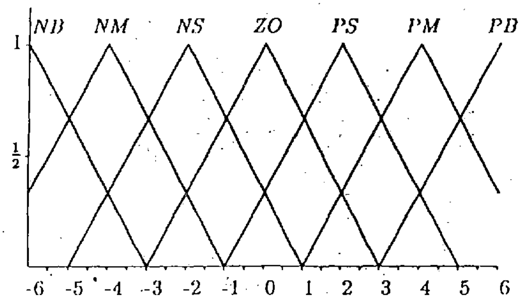


Figure 3.2 The fuzzy sets *NB*, *NM*, *NS*, *ZO*, *PS*, *PM* and *PB* on the domain $[-6, 6]$.

obtained, stored with minimal use of memory and manipulated efficiently by the inference engine.

The parametric, functional description of triangular shaped membership function is the most economic one. This explains the predominant use of this type of membership function. After selecting the shape of the membership function, each element of the term set is mapped on the domain of the corresponding linguistic variable. For example, this mapping in the case of e and ε=[-6,6] would be as shown in fig.3.2.

(ii) Choice of scaling factor

The use of normalized domain (universe of discourse) requires a scale transformation, which maps the physical values of the process state variable into a normalized domain. This is called input normalization, also output denormalization maps the normalized value of the control output variable into their respective physical domain.

The scaling factors which describe the particular input normalization and output denormalization play a role similar to that of the gain coefficient in a conventional controller.

They are utmost importance with respect to controller performance and stability related issues i.e. they are the source of possible instabilities, oscillation problem and deteriorated damping effect.

(i) Choice of variable and content of rules

Depending upon the type of controller to be designed P, PD, PI or PID like FKBC the choice of variables are process states and control output as well as the content of the rule antecedent and rule consequent for each of the rules.

The various notations used as

- error, denoted by e,
- change of error, denoted by Δe or $\dot{e}$
- Sum of errors, denoted by δe
- Change of control output, denoted by Δu or $\dot{u}$
- Control output, denoted by u

The analogy with a conventional controller we have

$$e[k] = y_{sp} - y[k]$$

$$\Delta e[k] = e[k] - e[k-1]$$

$$\Delta u[k] = u[k] - u[k-1]$$

where y_{sp} stands for set point and k is the sampling time

Here some choices of variable and content of rules for various controllers are discussed.

PD like FKBC

The conventional PD controller can be expressed by

$$u = K_P \,.\, e + K_D \,.\, \dot{e}\,.$$

K_P and K_D are proportional and differential gains

Then a PD like FKBC consist of rules is give below :

IF e[k] is < property symbol > AND Δe[k] is< property symbol > THEN u[k] is <property symbol >

when <property symbol> is the symbolic representation of linguistic variables.

PI like FKBC

The conventional PI controller can be expressed by

$$u = K_P \,.\, e + K_I \,.\, \int edt$$

where K_P and K_I are proportional and integral constant gain coefficient. Derivative of the above is given by

$$\dot{u} = k_P \,.\, \dot{e} + K_I \,.\, e$$

Then the rule for FKBC has the form

IF e is <Property symbol> AND De is <Property symbol> THEN Du is <Property symbol>

P-like FKBC

The symbolic representation of a rule for a P-like FKBC is given as

IF e is <Property symbol> THEN u is <Property symbol>

PID like FKBC

The equation describing a conventional PID controller is given as

$$u = K_P \,.\, e + k_D \,.\, \dot{e} + K_I \int e\,dt$$

The symbolic representation of the rules of a PID like FKBC is

IF e is <Property symbol> AND De is <Property symbol> AND Se is <Property symbol)

THEN u is <Property symbol>

(ii) *Choice of term set*

The term set σX of a linguistic variable X is described as consisting of a finite number of verbally (linguistically) expressed values which X can take. The linguistic value, members of the term set, are expressed as tuples of the form <value sign, value magnitude>, e.g. <positive big>, <negative big>, <negative small< etc., the value sign component of such a tuple takes on either one of the following two values, positive or negative. The value magnitude component can take on any number of linguistically expressed magnitude, e.g. {zero, small, medium, big} or {zero, small, big}.

(iii) *Derivation of rules*

There are three major approaches to the derivation of the rules of the FKBC

- ***Approach 1*** : This approach is the one that is most widely used today. It is based on the derivation of rules from the experience-based knowledge of the process operator or control engineer.

- ***Approach 2*** : This approach uses a linguistic description, viewed as a fuzzy model of the process under control to derive the set of rules of a FKBC.

- ***Approach 3*** : This approach, relies on the existence of a conventional process model usually a non-linear one. A well developed formal technique which uses a "fuzzy" version of the sliding mode control.

3.3 INFERENCE ENGINE

The inference engine or rule firing can be of two basic types [2] :

(i) Composition based inference

In this case, the fuzzy relation representing the meaning of each individual rule are aggregated into one fuzzy relation describing the meaning of the overall set of rules. Then inference or firing with this fuzzy relation is performed via the operation composition between the fuzzified crisp input and the fuzzy relation representing the meaning of the overall set of rules. As a result of the composition one obtains the fuzzy set describing the fuzzy value of the overall control output.

(ii) Individual rule based inference

In this case, first each single rule is fired. This firing can be simply described by (a) computing the degree of match between the crisp input and the fuzzy sets describing the meaning of the rule antecedent and (b) "Clipping" the fuzzy set describing the meaning of the rule consequent to the degree to which the rule antecedent has been matched by the crisp input finally the clipped values of the control output of each rule are aggregated, thus forming the value of the overall control.

3.4 DEFUZZIFICATION MODULE

The function of defuzzification Module (DM) are as follows [2, 4] :

3.4.1 DM-F1

It performs the so called defuzzification which converts the set of modified control output value into a single point wise values.

3.4.2 DM-F2

It performs the output denormalization which maps the pointwise value of the control output on to its physical domain. DM-F2 is not needed if non normalized domain are used. The design parameter of defuzzification module the choice of defuzzification method.

There are many defuzzification methods. They are

(i) Center of area/gravity defuzzification

(ii) Center of sums defuzzification

(iii) Center of largest area defuzzification

(iv) First of maximum defuzzification

(v) Middle of maximum defuzzification

(vi) Height defuzzification

Among the above the important one is center of gravity method.

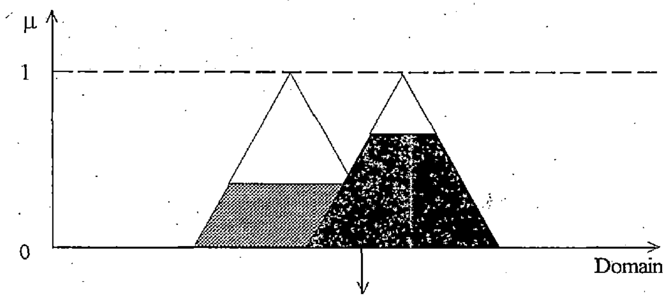


Fig.3.3 : A graphical representation of the centre of area defuzzification method

Center of gravity method

This is the best known defuzzification method, in descrete case ($u_l=\{u_1, u_2,\ldots,u_i\}$). This results in

$$u^* = \frac{\sum_{i=1}^{i} u_i \mu_u(u_i)}{\sum_{i=1}^{i} \mu_u(u_i)}$$

So this method determines the centre of the area below the combined membership function. Fig.3.3 shows the operation in graphical way.

3.5 FUZZY CONTROLLER OPERATION AN OVERVIEW

Let for any system the fuzzy set for error and error derivative shown in fig.3.4 and fuzzy associative memory are shown in fig.3.5.

Let normalized error and error change = 0.6 and 0.8

Ist Step : Find the fuzzy set for error and error derivative

fuzzy sets for error	=	4 and 5
fuzzy sets for error change	=	4 and 5

IInd Step : Find the height corresponding to each fuzzy set as shown in fig.3.6.

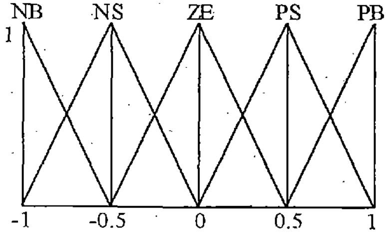


Fig.3.4 : Fuzzy sets for error, error derivative and control

$\dot{e}$ \ e	1	2	3	4	5
1	1	2	2	2	3
2	2	2	2	2	3
3	3	3	2	4	3
4	2	2	4	4	5
5	3	4	4	5	5

NB - 1
NS - 2
ZE - 3
PS - 4
PB - 5

Fig.3.5 : Fuzzy associated memory

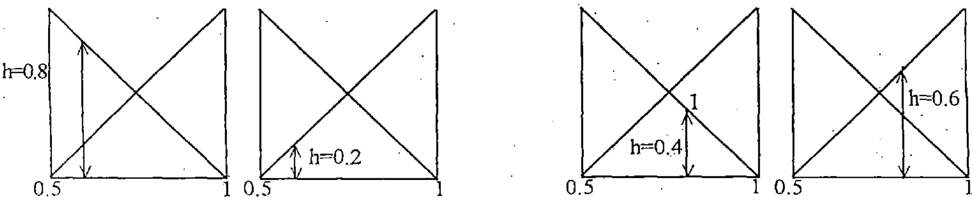


Fig.3.6 : Finding the height of fuzzy sets

IIIrd Step : Compare the heights of the above fuzzy set results

ai[j] compare bi[j]

min[1] = 0.4

min[2] = 0.6

min[3] = 0.2

min[4] = 0.2

IVth Step : Find the fuzzy set to be fired

iden[k] = fam [a1[i]] [a2[j]]

we get

iden[1] = 4,

iden[2] = 5

iden[3] = 5

iden[4] = 5

Vth Step : Arrange the above value by comparing to each other, we get

iden[1] = 4, iden[2] = 4, iden[3] = 4, iden[4] = 5

min[1] = 0.4, min[2] = 0.2, min[3] = 0.4, min[4] = 0.2

VIth Step : By center of gravity method

min[k] * a[3][4] = 0.5 x 0.4

min[k] * a[3][4] = 0.5 x 0.2

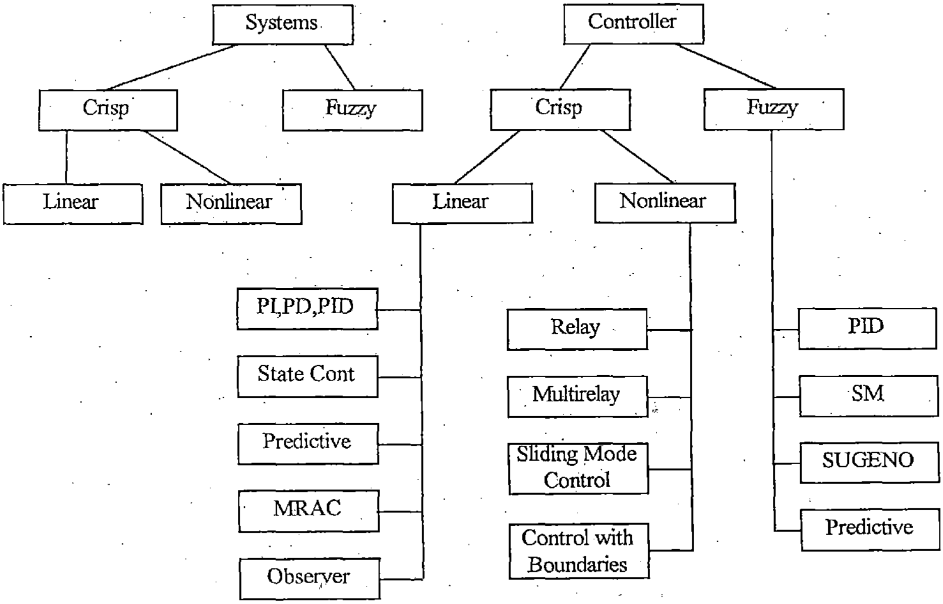


Fig.3.7 : An "Open scheme" of systems and controllers

min[k] * a[3][4] = 0.5 x 0.2

min[k] * a[3][4] = 0.1 x 0.2

$$\text{Control value } u = \frac{0.5 * 0.4 * 0.2 + 0.5 * 0.2 + 1 * 0.2}{0.4 + 0.2 + 0.2 + 0.2} = \frac{0.2 + 0.1 + 0.1 + 0.2}{1}$$

Control value (u) = 0.6

3.6 NON-LINEAR FUZZY CONTROL

The analytic functions in models of linear and non-linear system operate on the domain of crisp reals. In addition, we have the class of fuzzy systems whose models, in general, are algebraic mapping from the domain of crisp reals into a pre-specified domain of fuzzy reals.

The class of controller can be divided into linear, non-linear, and fuzzy knowledge base controllers.

Fig.3.7 shows an "open scheme" of systems and controllers.

The controller mentioned above uses two major knowledge's sources. The process operators or control engineers heuristic knowledge about the process and/or controller. In this case the model of the process/controller is described in terms of production rules

or if then rules only.

ERRATA

Page No.	Line No.	Typed	Correction
(ii)	4	envouragement	encouragement
(ii)	7	Wishes	wisher
(ii)	9	working	work
2	12	two	too
5	11	$u = =-2\ sgn(x_2)x_2-4x_1$	$u = -2sgn(x_2)x_2-4x_1$
21	-	21	24A
34	16	min[3] = 0.4	min[3] = 0.2
36	3	$\frac{0.5x0.4x0.2 + 0.5x0.2 + 1x0.2}{0.4 + 0.2 + 0.2 + 0.2}$	$\frac{0.5x0.4 + 0.5x0.2 + 0.5x0.2 + 1x0.2}{0.4 + 0.2 + 0.2 + 0.2}$
37	9	Structure 3.1	Structure 3.7
43	4	pu[i]=pu(i)-u[(i-1]	pu[i]=pu(i)-pu[(i-1)]
48	12	$x_2[t]=e^{at}[c_2((acosbt)+bcos(bt))$	$x_2[t]=e^{at}(c_2(bcosbt+asinbt)$
		$+ c_1((bsin(bt) - asin(bt))]$	$+ c_1(acosbt - bsinbt))$
49	11	Fuzzy_ctrl	fuzzctrl_in
52	Flow Chart	Fuzz_ctrl	fuzzctrl_in
67	8	ktu	khr

The non fuzzy model of the process (e.g. the phase plane of a second order system)

The general control law design principals are the same as in the case of crisp linear and non-linear systems.

1. Stability analysis
2. Performance analysis according to selected criteria
3. Rebustness analysis concerning parameter fluctuations, model uncertainties.

General design rules for designing the controller of structure 3.1

1. **Qualitative (symbolic) design of if than rules**

Defining the linguistic term set for the process state and control output variables and the corresponding membership functions describing the meaning of the element of these term sets.

Formulation of the set of IF THEN rules

2. **Quantitative design of scaling or normalisation factor this include the following steps.**

- Testing the system to be controlled with respect to controllably and observability.

- Analysis of the operating points and operation area of the crisp. process states, process outputs, and control variables.
- Utilization of design methods with origins in non-linear system theory

3.6.1 Sliding Mode FKBC

For a large class of non-linear systems FKBC are designed with respect to phase plane determined by error e and change of error $\dot{e}$ with respect to the states x and $\dot{x}$ [3,7]. A fuzzy value for the control variable is determined according to fuzzy values of error and change of error. The general approach to control design is the division of the phase plane into two semi planes by means of a switching line. Within the semi planes positive and negative control outputs are produced. The magnitude of the control output depends on the distance of the state vector from the switching line.

For a specific class of non-linear systems there is an appropriate robust control method called sliding mode control. The sliding mode control is especially appropriate for the tracking control of robot manipulator and also for motor whose mechanical load change over a wide range [2]

Let $x^{(n)} = f(x,t) + u + d$

where

$$x = (x, \dot{x}_1, \ldots, x^{(n-1)})^T$$

Furthermore, let Δf,d and $x_d^{(n)}$ have upper bound with known value $\tilde{F}$, D and v :

$$|\Delta f| \le \tilde{F}(x,t);\ |d| \le D(x,t);\ |x_d^{(n)}| \le v$$

The control problem is to obtain the state x for tracking a desired state x_d in the presence of model uncertainities and disturbances with the tracking error

$$e = x - x_d = (e,\ \dot{e},\dots,e^{(n-1)})^T$$

a stable switching surface is defined as follows

$$S(x,t) = 0$$

$$S(x,t) = (d/dt + \lambda)^{n-1}\, e \qquad \lambda \ge 0$$

Sufficient condition for the behaviour of sliding mode is

$$\frac{1}{2}\frac{d}{dt}(S^2(x,t) \le -\eta|s|. \qquad \eta \ge 0.$$

To achieve the sliding mode we choose u so that

$$u = (-\hat{f} - \lambda\dot{e}) - k(x,t),\ \mathrm{sgn}(s) \qquad \text{with } K(x,t) > 0$$

where $(-\hat{f} - \lambda\dot{e})$ is a compensation term and second term is the controller.

To avoid drastic changes of the control variables we substitute the function sgn(s) by sat(s/ϕ) where

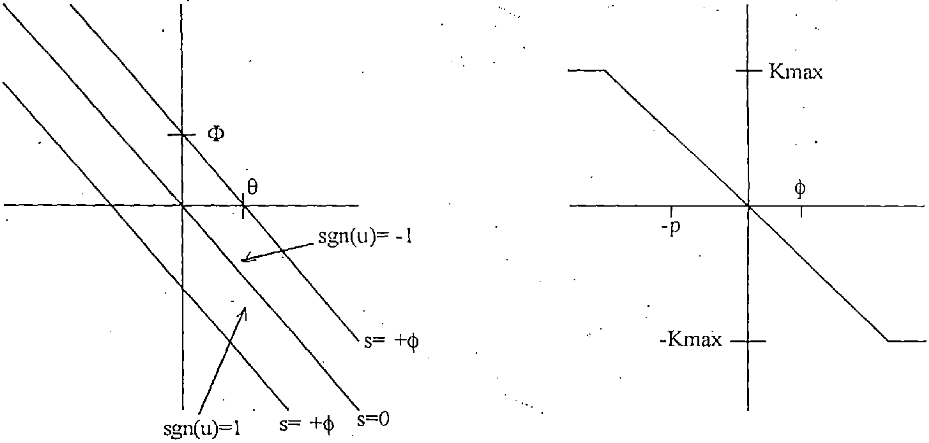


Fig.3.8 : Sliding mode principle with boundary layer

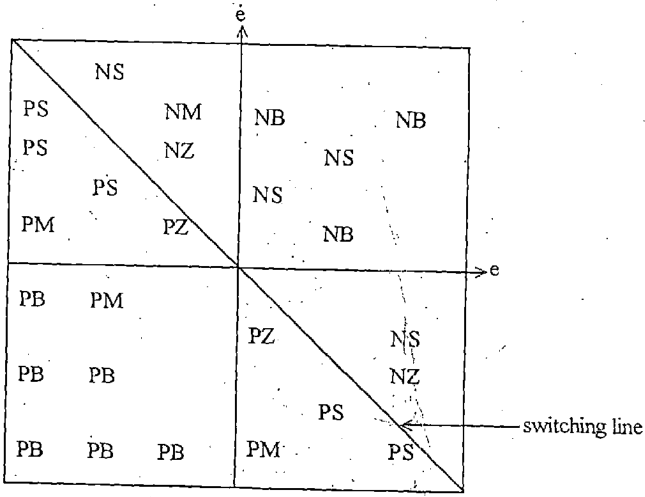


Fig. 3.9 : Rules in the normalized phase plane

$$\text{sat}(x) = \begin{cases} x & \text{if } |x| < 1 \\ \text{sgn}(s) & \text{if } |x| \geq 1 \end{cases}$$

As shown in fig.3.8, and the rules in the normalized phase plane is shown in fig.3.9. The working principle of a FKBC can be represented by

$$u = -k_{fuzz}(e, \dot{e}, \lambda)\ \text{sgn}(s)$$

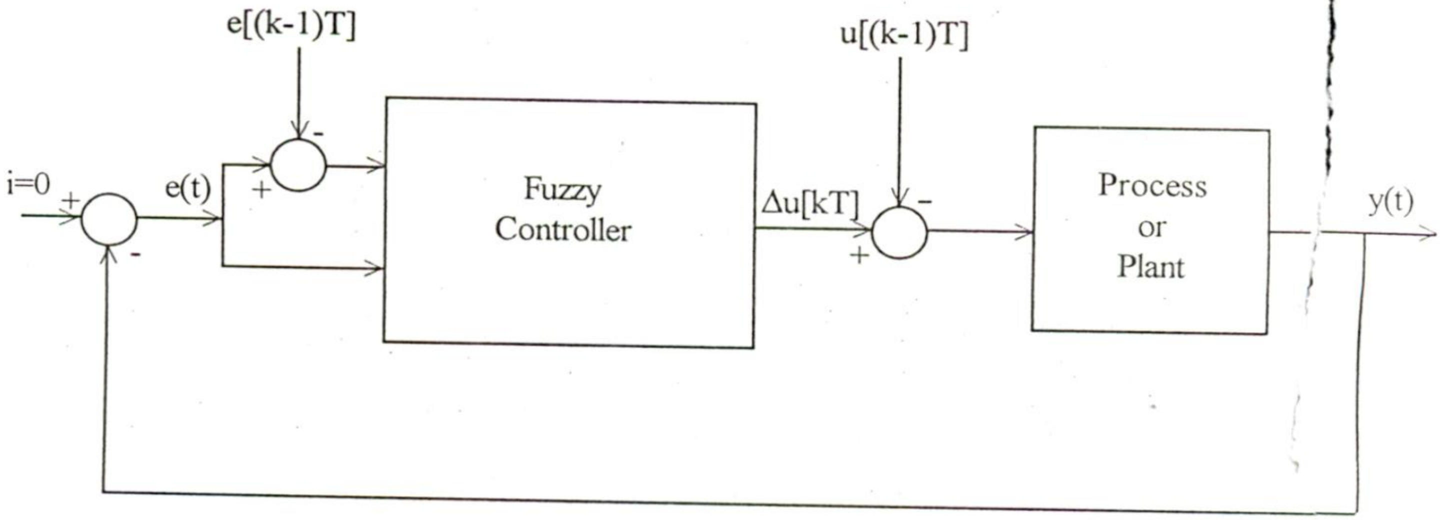


Fig.4.1 : Fuzzy controlled plant

CHAPTER - 4

DESIGN CONSIDERATIONS

4.1 FUZZY CONTROLLER

The fuzzy controller developed in this dissertation is to regulate the switching line having error, $e[i]=x_1[i]$ and change of error $\Delta e[i]=x_2[i]=\dot{x}_1[i]$ as the input variables and pu[i]=pu[i]-u[(i-1] which is the change in slope is the output variable as shown in fig.4.1 and i is the sampling time.

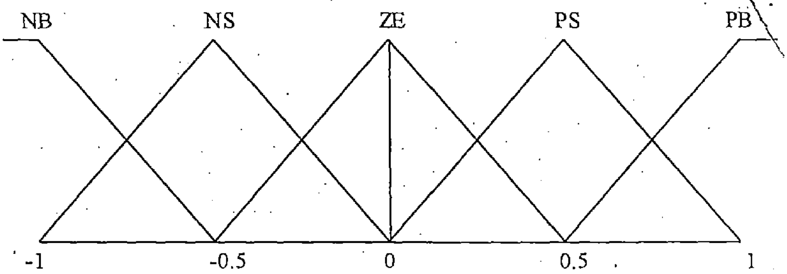
The membership function used for error(e), error change (Δe) and change of control (Δu) are the most natural and unbiased membership function i.e. symmetric triangle with equal bases and 50% overlap with the neighboring membership function as this provide significantly less reaching time. The term set of e, Δe and Δu contains five members, i.e.

{NB, NS, ZE, PS, PB}

where

NB = Negative big

NS = Negative small



Membership functions used in the fuzzy sets describing e, de, du

Fig.4.2 : Memberships of fuzzy sets used in the control rules

de \ e	NB	NS	ZE	PS	PB
NB	PB	PB	PB	PS	ZE
NS	PB	PS	PS	ZE	NS
ZE	PS	PS	ZE	NS	NS
PS	PB	ZE	NS	NS	NB
PB	ZE	NS	NB	NB	NB

Group 1

Group 2

Group 3

Group 4

Group 5

Fig.4.3 : Rule base

ZE	=	Zero
PS	=	Positive small
PB	=	Positive big

These five membership functions are distributed over the normalized domain (universe of discourse) [-1,1] as shown in fig.4.2. Membership function used in the fuzzy sets describing e, de, du:

Rule base for the fuzzy controller is shown in fig.4.3. The 25 entries of the table are the change of controller output (Δu) of fuzzy controller from the table justification of the rules can be given as follows :

Group-1 :

In this group of rules both error and error derivative are nearby equal i.e. either they are positive or negative equal or the error and error derivative are very close to switching line so control is almost zero.

Group-2 :

In this group the error and error derivative are above the switching line but not too away from it so the control required are negative small. The control required below the switching line are always negative.

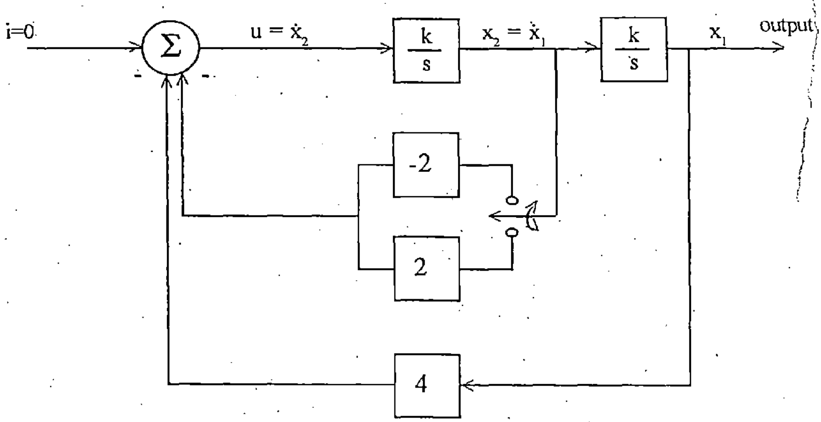


Fig.4.4 : Schematic diagram of controlled process

Group-3 :

In this group the error and error derivative are above the switching line and far from the switching line and the control required is negative big.

Group-4 :

In this group the error and error derivative are below the switching line but not far from it and the control required is positive small.

Group-5 :

In this group the error and error derivative are below the switching line and far from switching line so control is positive big.

4.2 PROCESS

The second order dynamical system shown in fig.4.4.

$$\dot{x}_1 = kx_2$$

$$\dot{x}_2 = ku \tag{4.1}$$

On simplification this gives

$$\dot{x}_1 = kx_2$$

$$\dot{x}_2 = = -2kx_2 \, \mathrm{sgn}(x_2) - 4kx_1, \tag{4.2}$$

The state space representation is given by

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} 0 & k \\ -4k & -2k\,\mathrm{sgn}(x_2) \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \tag{4.3}$$

Taking k to be unity the dynamical equation becomes

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -4 & -2\,\mathrm{sgn}(x_2) \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \tag{4.3}$$

If the switch is at the upper position the dynamical equation

$$\dot{x}_1 = x_2$$

$$\dot{x}_2 = = -4x_1 - 2x_2 \tag{4.5}$$

and

$$\ddot{x}_1 + 2\dot{x}_1 + 4x_1 = 0 \tag{4.6}$$

The two states are

$$x_1[t] = e^{at}(c1 \cos bt + c2 \sin bt) \tag{4.7}$$

$$x_2[t] = e^{at}[c2\,((a\cos bt) + b\cos(bt)) + c1\,((b\sin(bt) - a\sin(bt))] \tag{4.8}$$

where a and b are real and imaginary part of complex conjugate pole.

If the switch is at lower position the dynamical equation becomes

$$\ddot{x}_1 - 2\dot{x}_1 + 4x_1 = 0 \tag{4.9}$$

and the solution is given by eqn.4.7 and 4.8.

The two state variables found in the above section are used to find the slope of switching line at every instant of time. Depending upon the location of the two state variables the slope of the switching line is decided by the location with respect to different side of switching line. Once the trajectory intersects the switching line, the system is continuously changes its structure to attain the sliding mode.

4.3 DESCRIPTION OF SOFTWATE

The software developed in C language. Some of its details are :

Fuzzy_ctrl

This function takes the normalized input of error and error change as input and gives fuzzy sets of change of control (height) i.e. this function performs all the operations of fuzzy controller except normalization and denormalization.

Normal

This function takes error and error change at every sampling interval as input and convert them to normalized form between [-1, 1].

Height

This function determines the height of fuzzy sets for crisp error and error change in rule firing.

Search

This function is a sub-function of fuzzy_ctrl and determines the fuzzy sets for crisp input of error and error change i.e. it helps in determining rule firing.

Arrangevals

This function helps in defuzzification. It helps for including maximum height of output clipped fuzzy sets, if two or more rules results in same output fuzzy set.

Fuzzgraph

This function draws the response of the fuzzy controlled system on the screen.

4.4 FLOW CHARTS

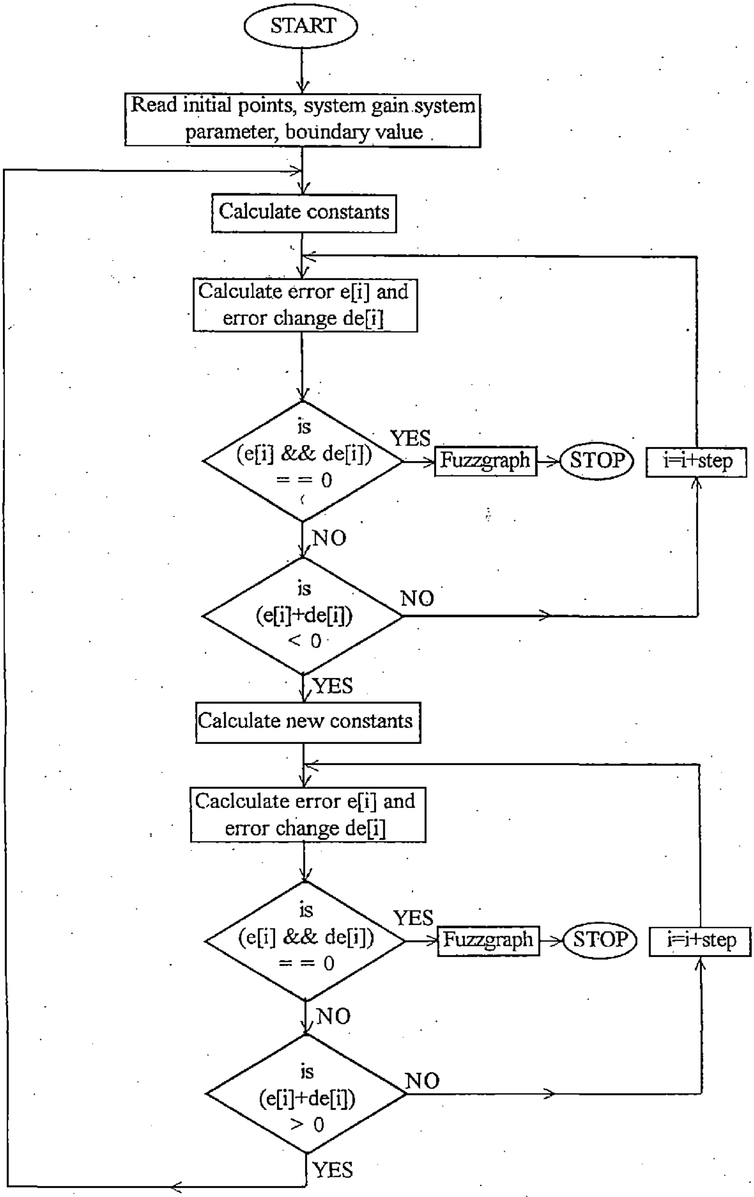


Fig.4.5 : Flow diagram of sliding mode control with fixed sliding surface

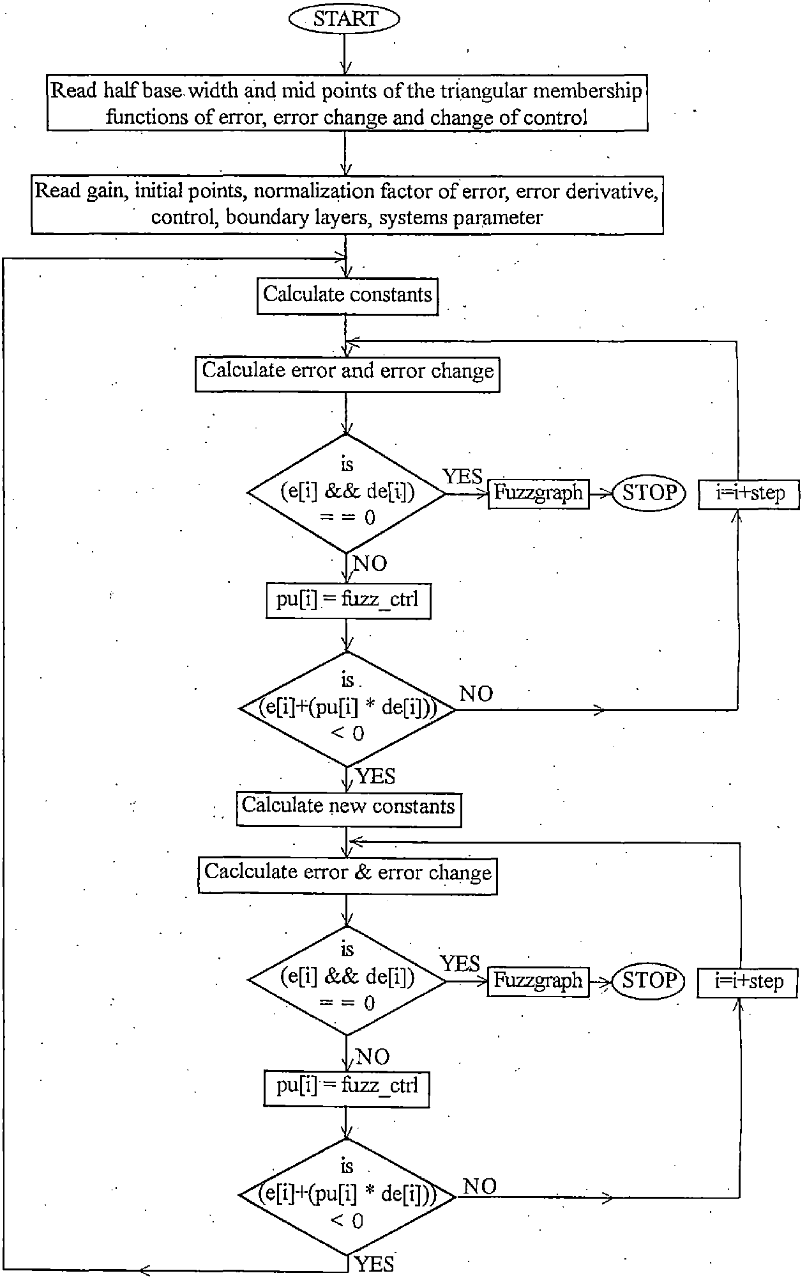


Fig.4.6 : Flow diagram of sliding mode control with fuzzy sliding surface

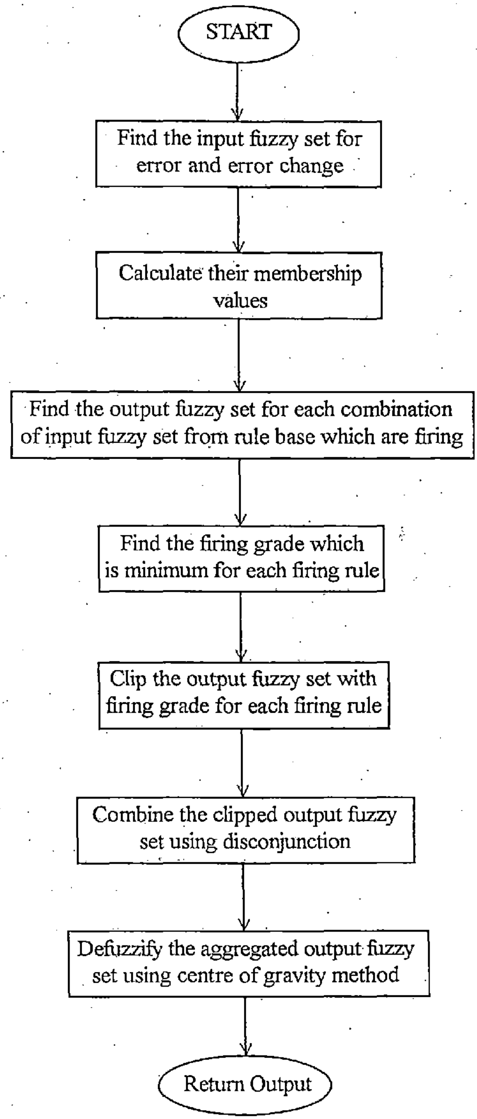


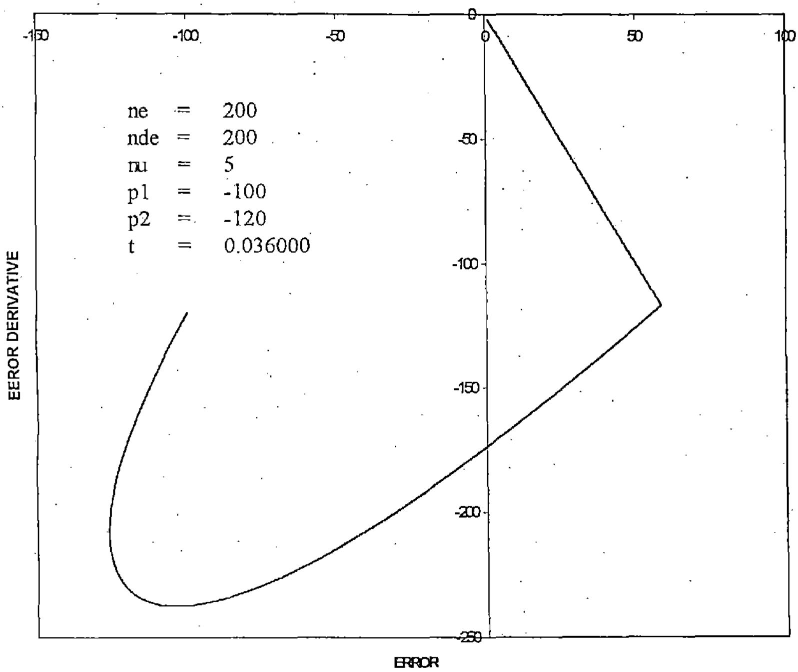
Fig.4.7 : Flow diagram of fuzz_ctrl subroutine

CHAPTER - 5

RESULTS AND DISCUSSIONS

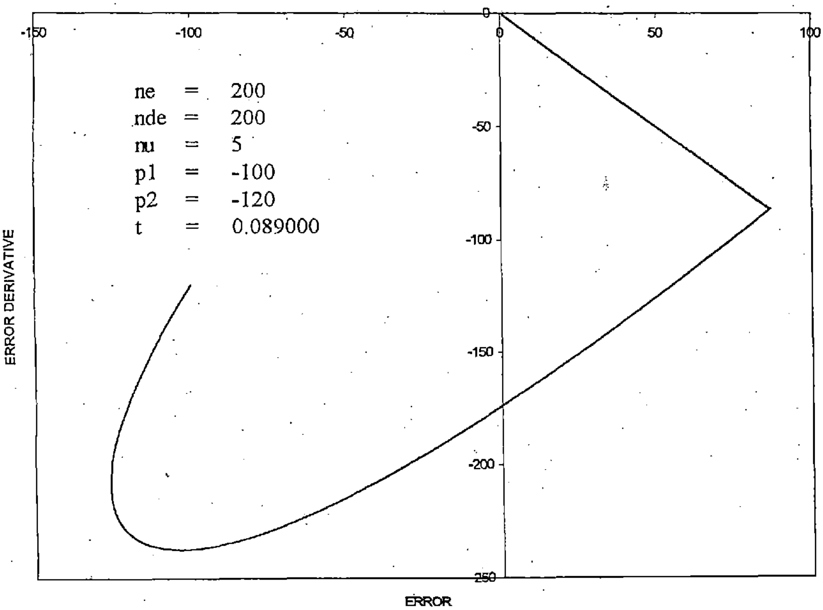
5.1 RESULTS

Response of system at initial tuning for fuzzy surface



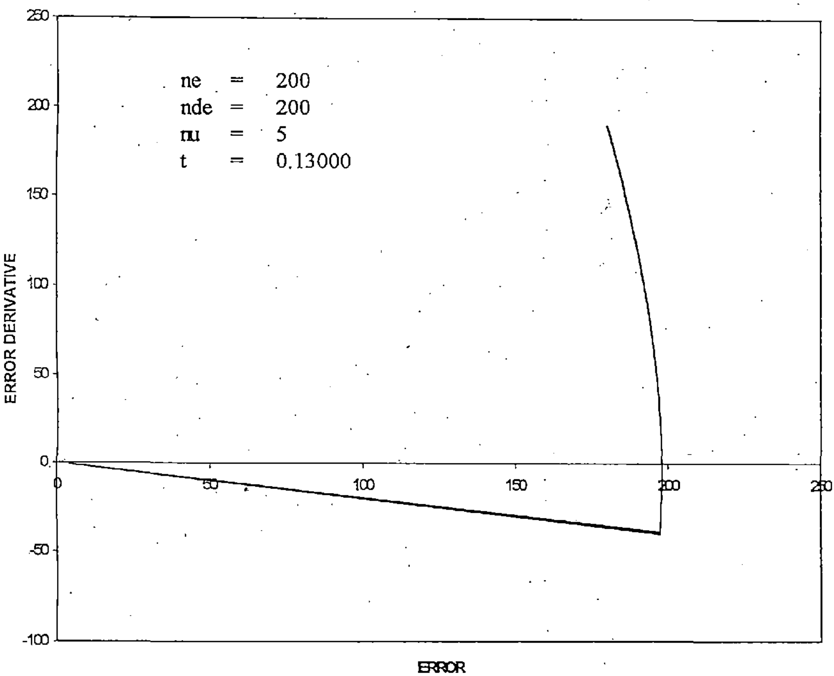
Graph 1

Response of system at initial tuning for fixed surface



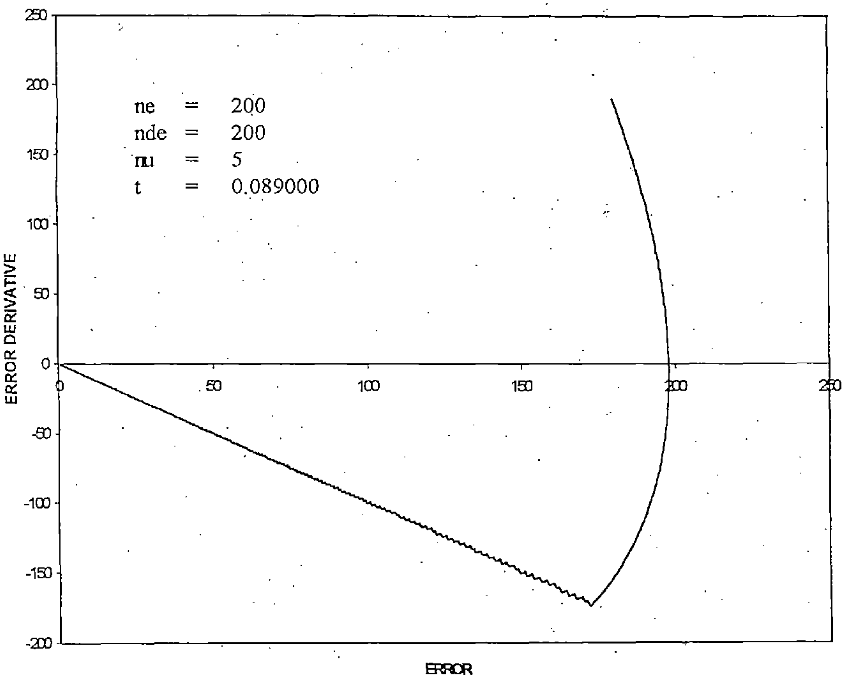
Graph 2

Effect of varying initial conditions



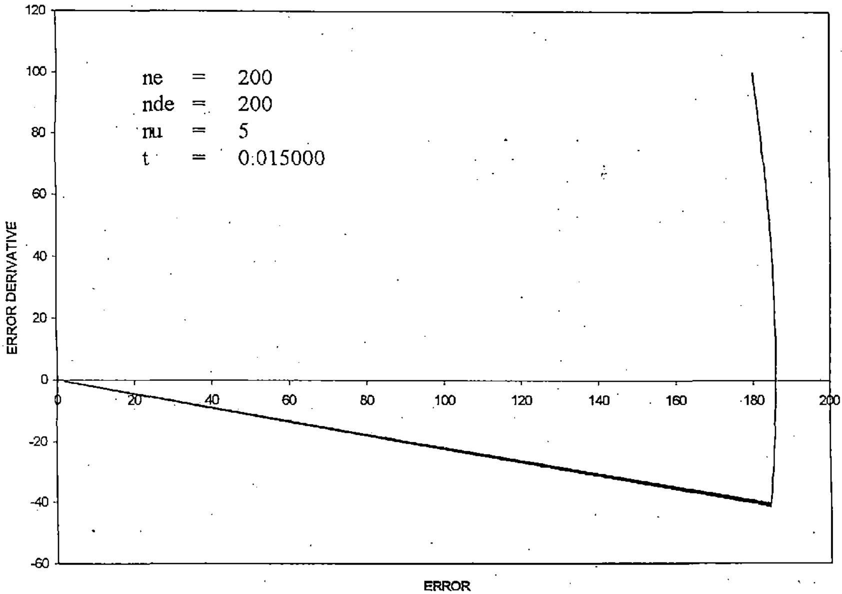
Graph 3 : Response of system having fuzzy surface with p1=180, p2=190

Effect of varying initial conditions



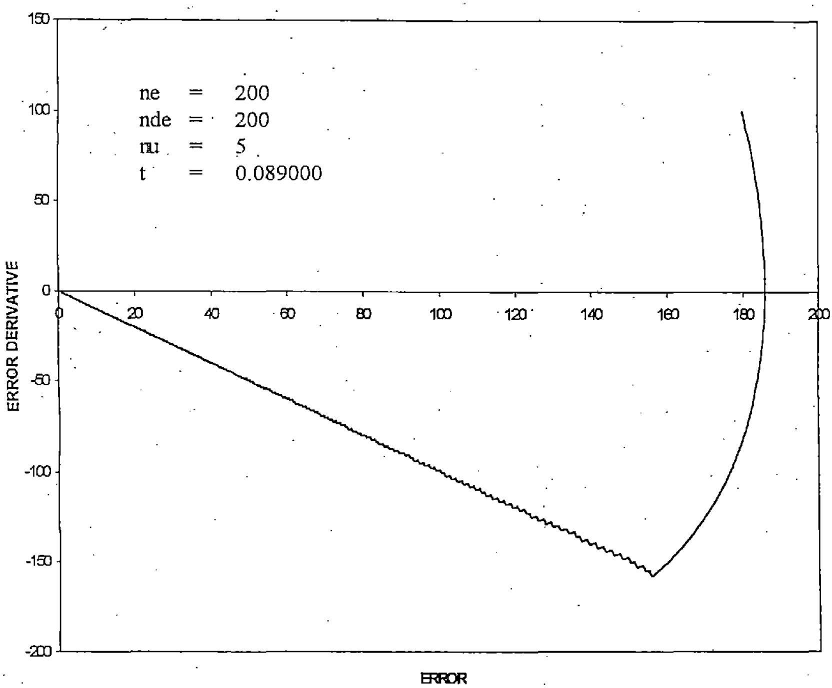
Graph 4 : Response of system having fixed surface with p1=180, p2=100

Effect of varying initial conditions



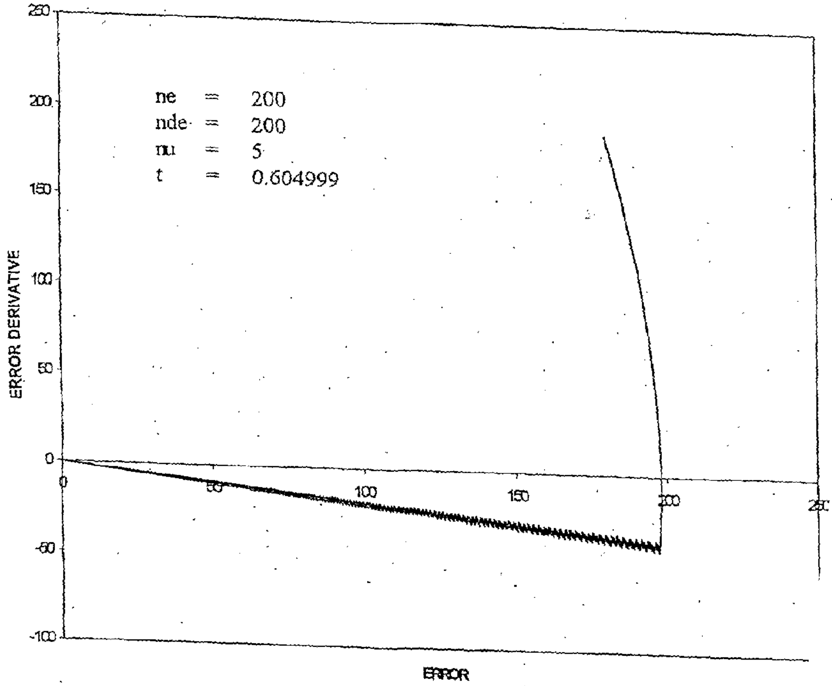
Graph 5 : Response of system having fuzzy surface with p1=180, p2=100

Effect of varying initial conditions



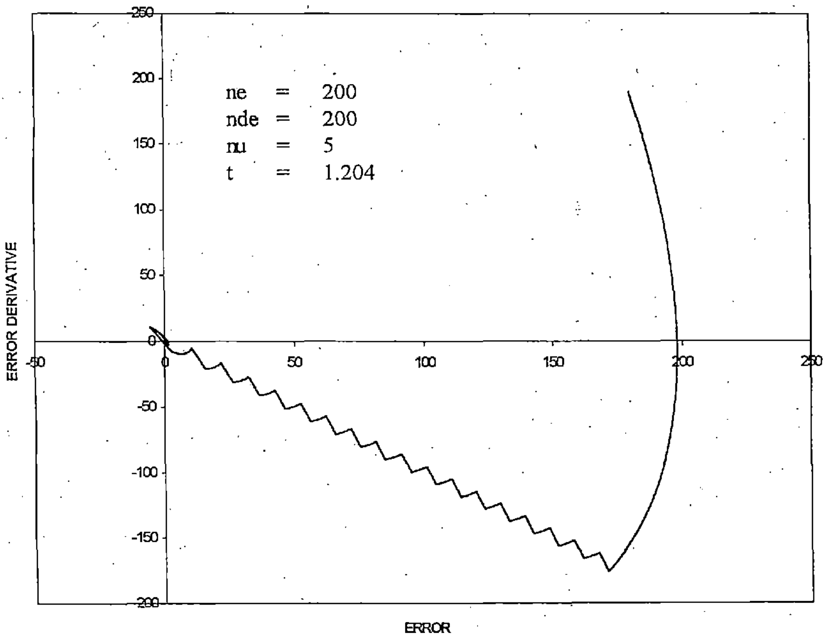
Graph 6 : Response of system having fixed surface with p1=180, p2=100

Effect of varying boundary value



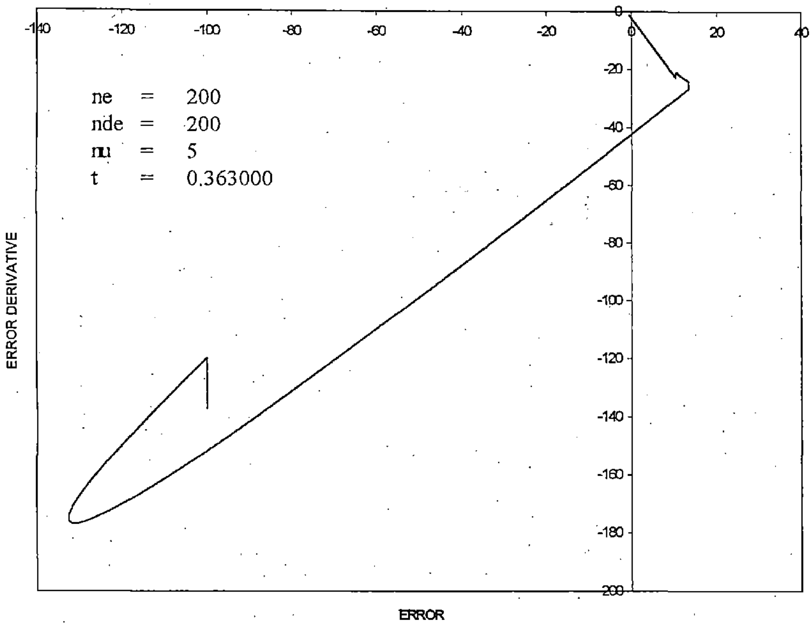
Graph 7 : Response of system having fuzzy surface with C5=5

Effect of varying boundary value



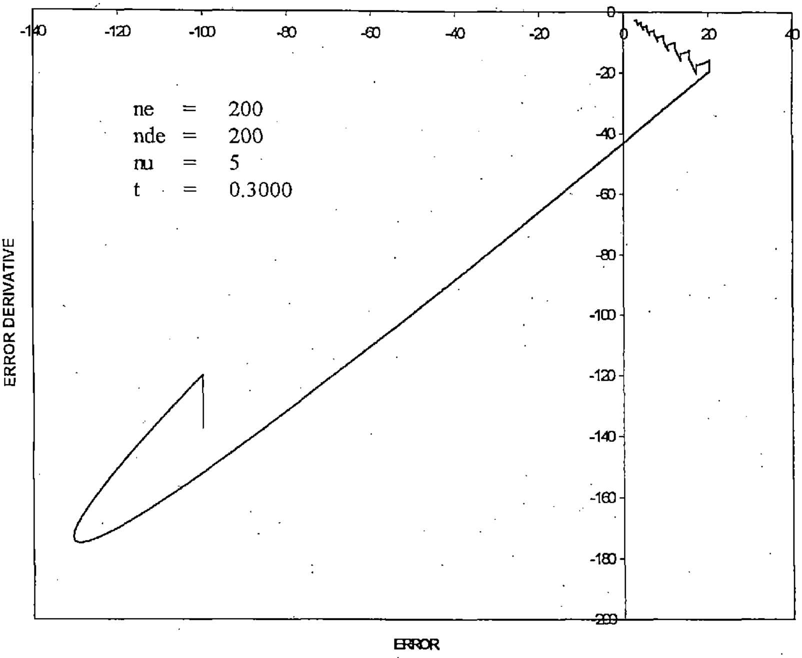
Graph 8 : Response of system having fixed surface with C5=5

Effect of varying system gain



Graph 9 : Response of system having fuzzy surface with K=1.1

Effect of varying system gain



Graph 10 : Response of system having fixed surface with K=1.1

5.2 DISCUSSIONS

Initially the parameters of the process i.e. gain (K), feedback gain and switched gain is taken as 1, ± 2 and 4 respectively.

By Nyquist stability criterion, process gain K was calculated for stability. The system is stable when the position of switch is at positive and unstable when the position of switch is at negative.

With the above value we found that both the systems i.e. with fixed and fuzzy surface exist sliding mode and the time require to attain the sliding mode is less in the case of fuzzy surface. At the above tuning the normalized phase plane is taken up to limit 200 i.e. ne=200, nde=200, the maximum angle of rotation of fuzzy surface is choosen as 33° negative and positive with respect to fixed switching time.

After the above tuning the system parameter is changed keeping scaling factors of fuzzy controller control. Following are the observation from the graph obtained after simulation :

(1) Graph 3, 4, 5, 6 shows the response of system with fixed and fuzzy surfaces from these figures we can see that the sliding mode can exists at every initial conditions in the normalized phase plane. Depending upon the initial conditions the slope of the fuzzy surface is high or less.

(2) Graph 7 and 8 show the effect of varying the boundary layer. The time required to reach the equilibrium point increases and both the systems

attain the sliding mode. The time require to attain the equilibrium points are 1.204 and 0.604999 respectively for fixed and fuzzy surfaces.

(3) Graph 9 and 10 show the effect of varying the systems gain K. From these graphs we can observe that the chattering is increases in both the case but the chattering is less in fuzzy surface then fixed surface.

CONCLUSIONS

From the responses obtained by simulating the second order system with fixed and fuzzy sliding surfaces, involving the variable structure, we conclude that the system having fuzzy surface require less time to attain the sliding mode at every initial conditions than fixed sliding surface. That is the system having fuzzy sliding surface is more superior than fixed surface. From graph 9 and 10 we conclude that if the gain of the system is increased the chattering is increase. Also from graph 7 and 8 we observe that increasing the boundary layer the time requires to reach the equilibrium points increases.

In this dissertation we have used triangular membership function for regulating the switching surface. In future work, the system can be simulated with bell-shaped membership function, which can smooth the fuzzy surface and can get better result than triangular membership function.

REFERENCES

1. Choi, S., Chear, C. and Park, D., "Moving switching surface for robust control of second order variable structure system", Int. J. Control, 1993, 58(1), pp.229-245.

2. D. Driankov, M. Reinfrank, "An induction to fuzzy control", Narosa Publication, 2nd Edition, 1997.

3. Decarlo, R.A., Zak, S.K. and Matthews, G.P., "Variable structure control of non-linear multivariable systems : a tutorial", Proc. IEEE, 1988, pp.212-232.

4. George J. Ktu/BO Yuan, "Fuzzy sets and fuzzy logic", Prentice Hall, 1977.

5. Gao, W. and Hung, J.C., "Variable structure control of non-linear systems : a new approach", IEEE Trans. Industrial Electronics, 1993, 40(1), pp.45-55.

6. Heejinlee, Euntaikimh Yung-Jinkang Mignonparks, "Design of a sliding mode controller with fuzzy sliding surfaces", IEEE Proc. Control Theory Appli., Vol.145, No.5, September 1998, pp.411-418.

7. Hung, J.Y., Gao, W. and Gao, J.C., "Variable structure control : a survey", IEEE Trans. Industrial Electronics, 1993, 40(1), pp.2-22.

8. Ishigame, A., Furukawa, T. and Kawamoto, S., "Sliding mode controller design based on fuzzy inference for non-linear systems", IEEE Trans. Industrial Electronics, 1993, 40(1), pp.64-70.

9. Moinuddin, "Design of robotic manipulator", Ph.D. Thesis Department of E & C, U.O.R., Roorkee, India, 1993, pp.8-35.

10. Slotine J.J. and Sastry, S.S., "Tracking control of non-linear systems using sliding surface with application to robot manipulator", Int. Journal of Control, 1983, 38(2), pp.465-492.

11. Timothy J. Ross, "Fuzzy logic with engineering application", Mc-Graw Hill Publication, 1997.

APPENDIX

```
/*PROGRAM FOR SLIDING MODE CONTRO WITH
  FIXED AND FUZZY SLIDING SURFACES*/
#include<stdio.h>
#include<stdlib.h>
#include<graphics.h>
#include<conio.h>
#include<math.h>
#include<dos.h>
#define SIZE 25
#define STEP .005
int iden[5],a1[3],a2[3],fam[6][6];
float a[4][6],b[4][6],c[4][6],min[5],normale,normalde;
float normalu,denormalu,p1,p2;

void main(void)
{
int d,z,choice;
int FSS;
float k,k1,i,j,p1,p2,p3,p4,c1,c2,c3,c4,c5,w=0,v=0,rx1,ix1,p,q,r,s,s1;
float t1,t2,t3;
float
e[SIZE],de[SIZE],ne[SIZE],nde[SIZE],du[SIZE],nu[SIZE],x1[SIZE],x2[SIZE]
,x3[SIZE],x4[SIZE],pu[SIZE],ku[SIZE];
extern int fam[6][6];
extern float a[4][6],b[4][6],c[4][6];
float normale=200,normalde=200,normalu=5,denormalu;
float normal(float,float);
float fuzzctrl_in(float,float);/*FUNCTIONS DEFINITION*/
void fuzzgraph(float);
FILE *P1;
P1=fopen("c2.dat","w");

a[1][1]=-1.0;a[1][2]=-0.5;a[1][3]=0.0;a[1][4]=0.5;a[1][5]=1.0;
a[2][1]=-1.0;a[2][2]=-0.5;a[2][3]=0.0;a[2][4]=0.5;a[2][5]=1.0;
a[3][1]=-1.0;a[3][2]=-0.5;a[3][3]=0.0;a[3][4]=0.5;a[3][5]=1.0;
b[1][1]=0.5;b[1][2]=0.5;b[1][3]=0.5;b[1][4]=0.5;b[1][5]=0.5;
b[2][1]=0.5;b[2][2]=0.5;b[2][3]=0.5;b[2][4]=0.5;b[2][5]=0.5;
b[3][1]=0.5;b[3][2]=0.5;b[3][3]=0.5;b[3][4]=0.5;b[3][5]=0.5;

fam[1][1]=5;fam[1][2]=5;fam[1][3]=5;fam[1][4]=4;fam[1][5]=3;
fam[2][1]=5;fam[2][2]=4;fam[2][3]=4;fam[2][4]=3;fam[2][5]=2;
fam[3][1]=4;fam[3][2]=4;fam[3][3]=3;fam[3][4]=2;fam[3][5]=2;
fam[4][1]=5;fam[4][2]=3;fam[4][3]=2;fam[4][4]=2;fam[4][5]=1;
fam[5][1]=3;fam[5][2]=2;fam[5][3]=1;fam[5][4]=1;fam[5][5]=1;


/*INPUT VARIABLES ENTERING*/
printf("FIXED SURFACE=1 \nFUZZY SURFACE=0\n");
scanf("%d",&FSS);
printf("initp1\n");
scanf("%f",&p1);
printf("initp2\n");
```

```
scanf("%f",&p2);
printf("p=");
scanf("%f",&p);
printf("q=");
scanf("%f",&q);
printf("r=");
scanf("%f",&r);
printf("k=");
scanf("%f",&k);/*GAIN*/

/* CALCULATION BEGINS HERE */
k1=k*k;
s=-(q*q-(4*p*r*k1));
s1=sqrt(s);
rx1=q/(2*p*k1);
ix1=(s1/(2*p*k1));
/*LOOP STARTED HERE*/

if(FSS==0){
nu[0]=fuzzctrl_in(p1/normale,p2/normalde);
if(nu[0]<0)
  pu[0]=-1*(nu[0]*normalu);
else
  pu[0]=(nu[0]*2.5)/normalu;}
else pu[0]=1;

c5=.1;/*BOUNDARY VALUE*/
c1=p1;
c2=(p2+p1)/ix1;
do
{
c1=p1;
c2=(p2+p1)/ix1;
for(i=0;i<SIZE;i+=STEP)
{
x1[i]=((c1*exp(-rx1*i)*cos(ix1*i))+(c2*exp(-rx1*i)*sin(ix1*i)));
e[i]=x1[i];
ne[i]=normal(e[i],normale);
x2[i]=(exp(-rx1*i)*((c1*(-rx1)*cos(ix1*i))+(c2*ix1*cos(ix1*i))
        +(c1*(-rx1)*sin(ix1*i))-(ix1*c2*sin(ix1*i))));
de[i]=x2[i];
nde[i]=normal(de[i],normalde);
nu[i]=fuzzctrl_in(ne[i],nde[i]);
du[i]=normalu*nu[i];
w+=du[i];
fprintf(P1,"%f\t%f\n",e[i],de[i]);
/*printf("%f;%f\n",e[i],de[i]); */
if((e[i]+(pu[i]*de[i]))>-c5)
continue;
else
p3=x1[i];
p4=x2[i];
t1=i;
break;
}
c3=p3;
c4=(p4-p3)/ix1;
```

```
for(j=0.0;j<SIZE;j+=STEP)
{
x3[j]=((c3*exp(rx1*j)*cos(ix1*j))+(c4*exp(rx1*j)*sin(ix1*j)));
e[j]=x3[j];
ne[j]=normal(e[j],normale);
x4[j]=(exp(rx1*j)*((c3*(rx1)*cos(ix1*j))+(c4*ix1*cos(ix1*j))
       +(c3*(rx1)*sin(ix1*j))-(ix1*c4*sin(ix1*j))));
de[j]=x4[j];
nde[j]=normal(de[j],normalde);
nu[j]=fuzzctrl_in(ne[j],nde[j]);
du[j]=normalu*nu[j];
v+=du[j];
fprintf(P1,"%f\t%f\n",e[j],de[j]);
if((e[j]+(pu[0]*de[j]))<c5)
continue;
else
p1=x3[j];
p2=x4[j];
t2=j;
break;
}
}
while((x1[i]>1&&x2[i]<1)||(x3[j]>1)&&(x4[j]<1));
t3=(t1+t2);printf("t3=%f",t3);getch();
fclose(P1);
fuzzgraph(SIZE);
}
/*FUZZYCONTROLLER SUBROUTINE*/
float fuzzctrl_in(float c,float d)
{
extern int iden[5],a1[3],a2[3],fam[6][6];
extern float min[5],a[4][6],b[4][6];
int i,j,l,k;
float error,error_change,sum1,sum2,p,u;
float memb1[3],memb2[3];
void search(float,float,float a[][6],float b[][6],int,int);
float height(float,float a[][6],float b[][6],int,int);
void arrangevals(void);
error=c;
error_change=d;
search(error,error_change,a,b,1,2);
for(i=1;i<=1;i++)
{
l=a1[i];
if(l==0)
memb1[i]=0.0;
else
memb1[i]=height(error,a,b,1,1);
memb1[2]=1-memb1[1];
}
for(i=1;i<=1;i++)
{
l=a2[i];
if(l==0)
memb2[i]=0.0;
else
memb2[i]=height(error_change,a,b,1,1);
```

```
memb2[2]=1-memb2[1];
}
if(memb1[1]>=memb2[1])
min[1]=memb2[1];
else
min[1]=memb1[1];
if(memb1[1]>=memb2[2])
min[2]=memb2[2];
else
min[2]=memb1[1];
if(memb1[2]>=memb2[1])
min[3]=memb2[1];
else
min[3]=memb1[2];
if(memb1[2]>memb2[2])
min[4]=memb2[2];
else
min[4]=memb1[2];

k=0;
for(i=1;i<=2;i++)
{
for(j=1;j<=2;j++)
{
++k;
if((a1[i])&&(a2[j])!=0)
iden[k]=fam[a1[i]][a2[j]];
}
}
/*defuzzification starts here*/
arrangevals() ;
sum1=0.0;
sum2=0.0;
for(k=1;k<=4;k++)
{
  if(k==1)
  {
sum1+=a[3][iden[k]]*min[k];
sum2+=min[k];
}
else
{
 if(iden[k]==iden[k-1]);
   {
    sum1+=a[3][iden[k]]*min[k];
   sum2+=min[k];
   }
   {
     p=(min[k]>min[k-1])?min[k]:min[k-1];
     sum1+=(a[3][iden[k]]*p)-a[3][iden[k]]*min[k-1];
     sum2+=p-min[k];
     }
} /*printf("sum1=%f sum2=%f",sum1,sum2);*/
} u=(sum1/sum2); if(u>1)u=1;
return(u);
}
```

```
void search(float v1 ,float v2 ,float a[][6],float b[][6],int n ,int m
)
{
extern int a1[3],a2[3];
int i,j,flag;
j=0,flag=0;
for(i=1;i<=5;i++)
{
if(i==1&&v1<a[1][1])
{
flag=1;
break;
}
if(i==5&&v1>a[1][5])
{
flag=2;
break;
}
if((v1>(a[n][i]-b[n][i]))&&(v1<=(a[n][i]+b[n][i]))){
++j;
 a1[j]=i;
 }
 }
if(flag==1)
{
a1[1]=1;
a1[2]=2;
}
if(flag==2)
{
a1[1]=5;
a1[2]=4;
}
flag=0;
j=0;
for(i=1;i<=5;i++)
     {
      if(i==1&&v2<a[2][1])
      {
flag=1;
break;
}
if(i==5&&v2>a[2][5])
{
flag=2;
break;
}
if((v2>(a[m][i]-b[m][i]))&&(v2<=(a[m][i]+b[m][i])))
{++j;
a2[j]=i;
}
}
if(flag==1){
a2[1]=1;
a2[2]=2;
}
if(flag==2){
```

```
a2[1]=5;
a2[2]=4;
}
return;
}
float height(float var,float a[][6],float b[][6],int n,int m)
{
float t,max;
if((n==1&&var<a[m][n])||(n==5&&var>a[m][n]))
return(1);
else if((n==2&&var<a[m][n])||(n==4&&var>a[m][n]))
return(0);
else
{
 if(var>0.0)
t=-(var+a[m][n]);
else
 t=(var-a[m][n]);
t=(t/b[m][n]);

if((1-t)>0)
max=1-t;
else max=t-1;
 } if(max>.5)
 max=1-max;
 return(max);
}
void arrangevals(void)
{
int i,j;
float temp,temp1;
extern int iden[5];
extern float min[5];
for(i=1;i<=4;i++)
   for(j=i+1;j<=4;j++)
     if(iden[i]>iden[j])
   {
      temp=iden[i];
      iden[i]=iden[j];
      iden[j]=temp;
   }
for(i=1;i<=4;i++)
  for(j=i+1;j<=4;j++)
  if(min[i]>min[j])
  {
   temp1=min[i];
   min[i]=min[j];
   min[j]=temp1;
   }
   return;
   }
float normal(float dn,float dnf)
{
float non;
non=dn/dnf;
return(non);
}
```

```
void fuzzgraph(float size)
{
float e[320],de[320],x1[320],x2[320];
float i,j;
FILE *P1;
int gd=DETECT,gm,x,y;
initgraph(&gd,&gm,"c\\tc\\prog\\tc\\bgi");
x=getmaxx();
y=getmaxy();
setcolor(WHITE);
P1=fopen("c2.dat","r");
for(i=0.0;i<=size;i+=STEP)
{
fscanf(P1,"%f\t%f\n",&e[i],&de[i]);
putpixel((x/2+e[i]),(y/2-de[i]),WHITE);
/*putpixel((i*10),(240+e[i]),WHITE);*/
line(0,y/2,x,y/2);
/*line(0,0,0,y);*/
line(x/2,0,x/2,y);
rectangle(10,10,100,80);
outtextxy(19,20,"ne=200");
outtextxy(19,30,"nde=200");
outtextxy(19,40,"nu=2");
outtextxy(19,50,"k=1");
outtextxy(19,60,"t=");
}
getch();
}
```