CHAPTER 4

CAN PROTOCOL DESIGN AND EXPERIMENTAL ANALYSIS

4.1 INTRODUCTION

The CAN is an ISO standard (ISO 11898) for serial communication. CAN is based on the “broadcast communication mechanism”, which is based on a message-oriented transmission protocol. It defines message contents rather than stations and station addresses. Every message has a message identifier, which is unique within the whole network since it defines content and also the priority of the message. This is important when several stations compete for bus access (bus arbitration). As a result of the content-oriented addressing scheme a high degree of system and configuration flexibility is achieved. It is easy to add stations to an existing CAN network without making any hardware or software modifications to the present stations as long as the new stations are purely receivers.
This allows for a modular concept and also permits the reception of multiple data and the synchronization of distributed processes. Also, data transmission is not based on the availability of specific types of stations, which allows simple servicing and upgrading of the network.

Controller Area Network or CAN protocol is a method of communication between various electronic devices like engine management systems, active suspension, ABS, gear control, lighting control, air conditioning, airbags, central locking etc. embedded in an automobile. An idea initiated by Robert Bosch GmbH in 1983 to improve the quality of automobiles thereby making them more reliable, safe and fuel efficient. With the developments taking place in the electronics and semiconductor industry the mechanical systems in an automobile were being replaced by more robust electronics system which had an improved performance.

New technologies, products and inventions with added or improved functions started to shape a complete new era for the automobile industry which promised more robust vehicles with use of electronics. The increasing number of electronic devices used communication signals with more complex interrelations between them. Thereby making the life difficult for automobile engineers when they designed systems wherein one electronic device needs to communicate with others to operate. Realizing the problem of communication between different electronic modules Robert Bosch came up with this new protocol called CAN which was first released in 1986. CAN provide a mechanism which is incorporated in the hardware and the software by which different electronic modules can communicate with each other using a common cable.

CAN protocol can be defined as the set of rules for transmitting and receiving messages in a network of electronic devices. It means that it defines how data is transferred from one device to another in a network. It was
designed specifically looking into the needs of the automobile industry. However CAN's robust architecture and advantages has forced many industries like Railway, Aircrafts, medical etc to adopt CAN protocol in their systems.

Every node has a Host controller (ECU/MCU) which is responsible for the functioning of the respective node. In addition to the host controller every node has a CAN controller and CAN transceiver. CAN controller convert the messages of the nodes in accordance with the CAN protocols to be transmitted via CAN transceiver over the serial bus and vice versa. CAN controller is a chip which can either be added separately or embedded inside the host controller of the node.

CAN does not follow the master-slave architecture which means every nodes has the access to read and write data on the CAN bus. When the node is ready to send data, it checks availability of the bus and writes a CAN frame onto the network. A frame is defined structure, carrying meaningful sequence of bit or bytes of data within the network. CAN transmitted frame does have address neither of transmitting node or the receiving node. CAN is a message based protocol. A message can be defined as a packet of data which carries information.

A CAN message is made up of 10 bytes of data. The data is organized in a specific structure called frame and the information carried in every byte is defined in the CAN protocol. Protocols are generally of two types: address based and message based. In an address based protocol the data packets contain the address of the destination device for which the message is intended. In a message based protocol every message is identified by a predefined unique ID rather than the destination addresses. All nodes on CAN receive the CAN frame and depending on ID on the node CAN decides whether to accept it or not. If multiple nodes send the message at the same
time than the node with highest priority (lowest arbitration ID) gets the bus access. Lower priority nodes wait till the bus is available.

- **Low cost:** As CAN serial bus uses two wires, it offers good price/performance ratio. Also, driven by high volume production of low cost protocol devices, they are relatively cheap.
- **Reliable:** Because of excellent error detection and error handling mechanisms used by CAN, it offers high reliability transmission. It is also highly immune to Electromagnetic Interference.
- **Flexibility:** CAN Nodes can be easily connected / disconnected. Also, the number of nodes is not limited by the protocol.
- **Good Speed:** CAN supports data rate of 1 MBit/s @ 40m bus length.
- **Multi-master communication:** Any node can access the bus.
- **Fault Confinement:** Faulty nodes do not disturb the communication.
- **Broadcast capability:** Messages can be sent to one /many/all nodes.
- **Standardized:** ISO has standardized CAN via ISO-DIS 11898 (high speed applications) and ISO-DIS 11519-2 (low speed applications). CAN protocol is also standardized by industry organizations such as SAE-Society of Automotive Engineers.

CAN is a low-level protocol, and does not support any security features intrinsically. Applications are expected to deploy their own security mechanisms to authenticate each other. Failure to do so may result in various sorts of attacks, if the opponent manages to insert messages on the us. Password mechanisms exist for data transfer that can modify the control.
unit software, like software download or ignition key codes, but usually not for standard communication.

CAN uses the existing OSI reference model to transfer data among nodes connected in a network. The OSI reference model defines a set of seven layers through which the data passes during communication between devices connected in a network. The 7-layered structure of the OSI model is a very robust approach widely adopted in many communication protocols. The figure below gives the clear picture of OSI model. Each layer has its specific function that supports the layer above and below as described under

- **Application layer**: It serves as a window for users and application processes to access network services. The common functions of the layers are resource sharing, remote file access, network management, electronic messages and so on.

- **Presentation layer**: The most important function of this layer is defining data formats such as ASCII text, EBCDIC text BINARY, BCD and JPEG. It acts as a translator for data into a format used by the application layer at the receiving end of the station.

- **Session layer**: It allows to establishing, communicating and terminating sessions between processes running on two different devices performing security, name recognition and logging.

- **Transport layer**: The transport layer ensures that messages are delivered error-free, in sequence, and without loss or duplication. It relieves the higher layer from any concern with the transfer of data between them and their peers.

- **Network layer**: It provides end to end logical addressing system so that a packet of data can be routed across several layers and establishes, connects and terminates network connections.
- Data link layer: It packages raw data into frames transferred from physical layer. This layer is responsible for transferring frames from one device to another without errors. After sending the frame it waits for the acknowledgement from receiving device. Data link layer has two sub layers:
  - MAC (Medium Access Control) layer: It performs frame coding, error detection, signaling, serialization and de-serialization.
  - LLC (Logical Link Control) layer: The LLC sub layer provides multiplexing mechanisms that make it possible for several network protocols (IP, Decnet and Appletalk) to coexist within a multipoint network and to be transported over the same network medium. It performs the function of multiplexing protocols transmitted by MAC layer while transmitting and decoding when receiving and providing node-to-node flow and error control.

The physical layer transmits bit from one device to another and regulates the transmission of bit streams. It defines the specific voltage and the type of cable to be used for transmission protocols. It provides the hardware means of sending and receiving data on a carrier defining cables, cards and physical aspects. CAN protocol uses lower two layers of OSI i.e. physical layer and data link layer. The remaining five layers that are communication layers are left out by BOSCH CAN specification for system designers to optimize and adapt according to their needs.

When developing and/or troubleshooting the CAN bus, examination of hardware signals can be very important. Logic analyzers and bus analyzers are tools which collect, analyse, decode and store signals so people can view the high-speed waveforms at their leisure. There are also specialist tools as well as CAN bus monitors.
4.2 REAL-TIME DATA TRANSMISSION

In real-time processing the urgency of messages to be exchanged over the network can differ greatly: a rapidly changing dimension, e.g. engine load, has to be transmitted more frequently and therefore with less delays than other dimensions, e.g. engine temperature. The priority, at which a message is transmitted compared to another less urgent message, is specified by the identifier of each message. The priorities are laid down during system design in the form of corresponding binary values and cannot be changed dynamically. The identifier with the lowest binary number has the highest priority.

Bus access conflicts are resolved by bit-wise arbitration of the identifiers involved by each station observing the bus level bit for bit. This happens in accordance with the wired-and-mechanism, by which the dominant state overwrites the recessive state. All those stations (nodes) with recessive transmission and dominant observation lose the competition for bus access. All those "losers" automatically become receivers of the message with the highest priority and do not re-attempt transmission until the bus is available again.

Transmission requests are handled in order of their importance for the system as a whole. This proves especially advantageous in overload situations. Since bus access is prioritized on the basis of the messages, it is possible to guarantee low individual latency times in real-time systems.
4.2.1 Message Frame Formats

The CAN protocol supports two message frame formats, the only essential difference being in the length of the identifier. The “CAN base frame” supports a length of 11 bits for the identifier, and the “CAN extended frame” supports a length of 29 bits for the identifier.
frame. The following "Control field" contains the "IDentifier Extension (IDE)" bit to distinguish between the CAN base frame and the CAN extended frame, as well as the "Data Length Code (DLC)" used to indicate the number of following data bytes in the "Data field". If the message is used as a remote frame, the DLC contains the number of requested data bytes.

The "Data field" that follows is able to hold up to 8 data byte. The integrity of the frame is guaranteed by the following "Cyclic Redundant Check (CRC)" sum. The "ACKnowledge (ACK) field" compromises the ACK slot and the ACK delimiter. The bit in the ACK slot is sent as a recessive bit and is overwritten as a dominant bit by those receivers, which have at this time received the data correctly. Correct messages are acknowledged by the receivers regardless of the result of the acceptance test. The end of the message is indicated by "End Of Frame (EOF)". The "Intermission Frame Space (IFS)" is the minimum number of bits separating consecutive messages. Unless another station starts transmitting, the bus remains idle after this.

4.3.1 CAN Extended Frame Format

The difference between an extended frame format message and a base frame format message is the length of the identifier used. The 29-bit identifier is made up of the 11-bit identifier ("base identifier") and an 18-bit extension ("identifier extension"). The distinction between CAN base frame format and CAN extended frame format is made by using the IDE bit, which is transmitted as dominant in case of an 11-bit frame, and transmitted as recessive in case of a 29-bit frame. As the two formats have to co-exist on one bus, it is laid down which message has higher priority on the bus in the case of bus access collision with different formats and the same identifier / base identifier.
The 11-bit message always has priority over the 29-bit message. The extended format has some trade-offs: The bus latency time is longer (in minimum 20 bit-times), messages in extended format require more bandwidth (about 20%), and the error detection performance is lower (because the chosen polynomial for the 15-bit CRC is optimized for frame length up to 112 bits).

CAN controllers, which support extended frame format messages are also able to send and receive messages in CAN base frame format. CAN controllers that just cover the base frame format do not interpret extended frames correctly. However there are CAN controllers, which only support the base frame format but recognize extended messages and ignore them.

4.3.2 Detecting and Signalling Errors

Unlike other bus systems, the CAN protocol does not use acknowledgement messages but instead signals errors immediately as they occur. For error detection the CAN protocol implements three mechanisms at the message level (data link layer: OSI layer 2):

Cyclic Redundancy Check (CRC): The CRC safeguards the information in the frame by adding a frame check sequence (FCS) at the transmission end. At the receiver this FCS is re-computed and tested against the received FCS. If they do not match, there has been a CRC error.

Frame check, This mechanism verifies the structure of the transmitted frame by checking the bit fields against the fixed format and the frame size. Errors detected by frame checks are designated "format errors". ACK errors: Receivers of a message acknowledge the received frames. If the transmitter does not receive an acknowledgement an ACK error is indicated.
The CAN protocol also implements two mechanisms for error detection at the bit level (physical layer: OSI layer 1): Monitoring: The ability of the transmitter to detect errors is based on the monitoring of bus signals. Each station that transmits also observes the bus level and thus detects differences between the bit sent and the bit received. This permits reliable detection of global errors and errors local to the transmitter.

Bit stuffing: The coding of the individual bits is tested at bit level. The bit representation used by CAN is "Non Return to Zero (NRZ)" coding. The synchronization edges are generated by means of bit stuffing. That means after five consecutive equal bits the transmitter inserts a stuff bit into the bit stream. This stuff bit has a complementary value, which is removed by the receivers.

If one or more errors are discovered by at least one station using the above mechanisms, the current transmission is aborted by sending an "error frame". This prevents other stations from accepting the message and thus ensures the consistency of data throughout the network. After transmission of an erroneous message that has been aborted, the sender automatically re-attempts transmission (automatic re-transmission). Nodes may again compete for bus access.

4.4 MODELLING OF DC AND INDUCTION MOTOR

DC motors, because of their simplicity, ease of application, reliability and favourable cost have been a backbone of industrial applications. In these applications, the motor should be precisely controlled to give the desired performance. Modelling of any system is an important task in control applications because the electrical and mechanical components should be represented in mathematical form. Although several methods are available for representing DC Motor model,
the state space approach is unique in the sense, accurate and precise control is possible.

The state space approach in modelling of DC Motor is designed and mathematical models are developed for simple closed loop control of DC Motor and closed loop control with proportional Integral and Derivative (PID) controller. This approach of modelling is useful for designing intelligent control techniques like Fuzzy control, neural networks etc.

In the recent years the usage of data networks has been increased due to its cost effective and flexible applications. A shared data network can effectively reduce complicated wiring connections installation and maintenance for connecting a complex control system with various sensors, actuators, and controllers as a networked control system. For the time-sensitive application with networked control system the remote dc motor actuation control has been chosen.

Due to time-varying network traffic demands and disturbances, the guarantee of transmitting signals without any delays or data losses plays a vital role for the performances in using networked control systems. This paper proposes Fuzzy Logic Controller methodology in the networked dc motor control and the results are compared with the performance of the system with Ziegler-Nichols Tuned Proportional-Integral-Derivative Controller and Fuzzy Modulated Proportional-Integral-Derivative Controller. Simulations results are presented to demonstrate the proposed schemes in a closed loop control. The effective results show that the performance of networked control dc motor is improved by using Fuzzy Logic Controller than the other controllers.
The adaptation of communication network for information exchange between controllers, sensors and actuators to realize a closed control loop is called as Networked Control System (NCS). Networks reduce the complexity in wiring connections and the costs of Medias; provide ease in maintenance and also enable remote data transfer and data exchanges among users. Therefore, NCS is used widely in many industrial applications. Two major challenges as networked induced delay and data losses in the network affects the performance of the system.

Hence the challenges have to be compensated. Thus, with a networked controlled dc motor system this paper illustrates the proposed Fuzzy Logic Controller (FLC) for the compensation of the challenges and also compares FLC simulation results with the Fuzzy Modulated Proportional-Integral-Derivative Controller (FMPID) and Zeigler Nichols tuned Proportional-Integral-Derivative (PID) Controller.

There are two approaches to utilize a data network as Hierarchical Structure and Direct Structure. The Hierarchical Structure is the dc motor is controlled by its own remote controller at remote station. The central controller provides the set point to the plant (dc motor) via remote controller and the sensor measurements of the system are sent from the remote station to central controller. The remote controller controls the plant by providing the control signal in the remote unit. The set points and sensor measurements are transmitted through network. This approach has a poor interaction between the central and remote unit because of not transmitting the control signal from central controller. Whereas in the Direct Structure approach the network is used for the direct transfer of the control signal and the sensor measurements between a remote unit and a central controller.
The central controller is connected to the dc motor through an interface unit. Due to the transfer of control signal directly to plant this approach provided better interaction of data’s between central controller and the plant than the hier archical structure.

Recently the stability analysis and control design for NCS have attracted considerable research interest. The presents an approach for stability analysis of NCS that decouples the scheduling protocol from properties of network free nominal closed-loop system. Nesic and Teel extended by stochastic deterministic protocols in the presence of random packet dropouts and inter transmission time and they also proposed wireless scheduling protocol for non-linear NCS.

In most of the advanced control algorithms DC Motors are used because of stable and linear characteristics associated with it. Also various speed control methods are available for DC Motor to meet the desired performance. Hence modelling should be done in such a way that every control algorithm available can be implemented. Generally modelling of any system is to represent mechanical, electrical or any physical systems or components in mathematical form.

The DC Motor can be modelled by using four basic equations. Regularly every physical system is represented in the form of transfer function which is a relation between input and output, but absence of initial conditions limits this form of modelling. DC Motor can also be modelled by using state space equations. The state space representation is relation between state variables, their derivatives, input and output. Feasibility of defining initial conditions and being simple first order differential equations the state space approach finds its application in modelling various physical systems.
The motor variables to be controlled are the speed and the armature current. In the proposed fuzzy logic control scheme, the motor speed error and the error change are used as input variables to Fuzzy speed controller. However, the armature current error and the error change are the input variables to fuzzy current controller. The error and error change for both speed and current are scaled using appropriate scaling factors.

These scaled input data are then converted into linguistic variables, which may be viewed as labels of fuzzy sets. The linguistic variables, which are used for the input variables. Also, the choice of membership function shape is mainly dependent on the designer preference. For simplicity, the trapezoidal and triangular shaped functions are used in this application. In the universe of discourse, the numbers for the aforementioned linguistic variables are selected.

Fuzzy logic and proportional-integral-derivative (PID) controllers are compared for use in direct current (DC) motors positioning system. A simulation study of the PID position controller for the armature-controlled with fixed field and field-controlled with fixed armature current DC motors is performed. Fuzzy rules and the inferencing mechanism of the fuzzy logic controller (FLC) are evaluated by using conventional rule-lookup tables that encode the control knowledge in a rules form.

The performance assessment of the studied position controllers is based on transient response and error integral criteria. The results obtained from the FLC are not only superior in the rise time, speed fluctuations, and percent overshoot but also much better in the controller output signal structure, which is much remarkable in terms of the hardware implementation.

Lotfi Zadeh, the father of fuzzy logic, claimed that many sets in the world that surrounds us are defined by a non-distinct boundary. Zadeh
decided to extend two-valued logic, defined by the binary pair \{0, 1\}, to the whole continuous interval \[0, 1\], thereby introducing a gradual transition from falsehood to truth.

Fuzzy control is a control method based on fuzzy logic. Just as fuzzy logic can be described simply as "computing with words rather than numbers"; fuzzy control can be described simply as "control with sentences rather than equations". A fuzzy controller can include empirical rules, and that is especially useful in operator controlled plants. A comprehensive review of the classical design and implementation of the fuzzy logic controller can be found in the literature.

A fuzzy IF-THEN rule-based system consists of the following modules. Fuzzification: Converting crisp facts into fuzzy sets described by linguistic expressions. Membership functions can be flat on the top, piece-wise linear and triangle shaped, rectangular, or ramps with horizontal shoulders.

The fuzzy IF-THEN rule expresses a fuzzy implication relation between the fuzzy sets of the premise and the fuzzy sets of the conclusion. The following steps describe this process:

- Matching of the facts with the rule premises (determination of the degree of firing DOF of the facts to the rule premises).
- If the rule contains logic connections between several premises by fuzzy AND or fuzzy OR the evaluation is performed by t-norms or t-conorms (the result gives then the DOF of the facts to the complete premise).
- The next step is the determination of the individual rule output. The DOF of a rule interacts with its consequent to provide the output of the rule. It will be a fuzzy subset over the output universe.
This process aggregates the individual rule outputs to obtain the overall system output. It will be also a fuzzy subset over the output universe (a union operation yields a global fuzzy set of the output).

Defuzzification to obtain crisp output (various defuzzification methods can be used, as, e.g., center of gravity, bisector of area, and mean of maximum, to obtain a crisp numerical output value).

A PID-like (proportional plus integral plus derivative, PID) fuzzy logic controller (FLC), or simply PID-like FLC, algorithms have been and continue to be a very active and fruitful research field since Mamdani and Assilian pioneering work on fuzzy controller.

The impetus behind this continuity of the subject lies largely in the fact the conventional PID algorithms has been successfully used in the process industries and remains the most commonly used algorithm today, while numerous application of fuzzy logic control (FLC) have emerged covering a wide range of practical areas [8] and that many software and hardware products for fuzzy control have been commercialized during the last few years.

4.4.1 DC Motor Model

In armature control of separately excited DC motors, the voltage applied to the armature of the motor is adjusted without changing the voltage applied to the field. Figure 4.4 shows a separately excited DC motor equivalent model.
where,

Ra = Armature Resistance

Va = Armature Voltage

La = Armature Inductance

Ia = Armature Current

Eb = Back emf

w = angular speed

Tm = Motor Torque

θ = Angular position of ‘r’.

The direct current motor or the DC motor has a lot of application in today’s field of engineering and technology. Starting from an electric shaver to parts of automobiles, in all small or medium sized motoring applications DC motors come handy. And because of its wide range of application different
functional types of dc motor are available in the market for specific requirements.

The types of DC motor can be listed as follows

• DC motor
  • Permanent Magnet DC Motor
  • Separately Excited DC Motor
  • Self Excited DC Motor
    • Shunt Wound DC Motor
    • Series Wound DC Motor
    • Compound Wound DC Motor
      • Cumulative compound DC motor
        • Short shunt DC Motor
        • Long shunt DC Motor
      • Differential Compound DC Motor
        • Short Shunt DC Motor
        • Long Shunt DC Motor

Other types of DC motors require no commutation.

• Homopolar motor – A homopolar motor has a magnetic field along the axis of rotation and an electric current that at some point is not parallel to the magnetic field. The name homopolar refers to the absence of polarity change. Homopolar motors necessarily have a single-turn coil, which limits them to very low voltages. This has restricted the practical application of this type of motor.

• Ball bearing motor – A ball bearing motor is an unusual electric motor that consists of two ball bearing-type bearings, with the inner races mounted on a common conductive shaft, and the outer races connected
to a high current, low voltage power supply. An alternative construction fits the outer races inside a metal tube, while the inner races are mounted on a shaft with a non-conductive section (e.g. two sleeves on an insulating rod). This method has the advantage that the tube will act as a flywheel. The direction of rotation is determined by the initial spin which is usually required to get it going.

A coil of wire with a current running through it generates an electromagnetic field aligned with the center of the coil. The direction and magnitude of the magnetic field produced by the coil can be changed with the direction and magnitude of the current flowing through it.

A simple DC motor has a stationary set of magnets in the stator and an armature with one more windings of insulated wire wrapped around a soft iron core that concentrates the magnetic field. The windings usually have multiple turns around the core, and in large motors there can be several parallel current paths. The ends of the wire winding are connected to a commutator. The commutator allows each armature coil to be energized in turn and connects the rotating coils with the external power supply through brushes. (Brushless DC motors have electronics that switch the DC current to each coil on and off and have no brushes.)

The total amount of current sent to the coil, the coil's size and what it's wrapped around dictate the strength of the electromagnetic field created. The sequence of turning a particular coil on or off dictates what direction the effective electromagnetic fields are pointed. By turning on and off coils in sequence a rotating magnetic field can be created. These rotating magnetic fields interact with the magnetic fields of the magnets (permanent or electromagnets) in the stationary part of the motor (stator) to create a force on the armature which causes it to rotate.
In some DC motor designs the stator fields use electromagnets to create their magnetic fields which allow greater control over the motor.

A PM motor does not have a field winding on the stator frame, instead relying on PMs to provide the magnetic field against which the rotor field interacts to produce torque. Compensating windings in series with the armature may be used on large motors to improve commutation under load. Because this field is fixed, it cannot be adjusted for speed control. PM fields (stators) are convenient in miniature motors to eliminate the power consumption of the field winding. Most larger DC motors are of the "dynamo" type, which have stator windings. Historically, PMs could not be made to retain high flux if they were disassembled; field windings were more practical to obtain the needed amount of flux. However, large PMs are costly, as well as dangerous and difficult to assemble; this favors wound fields for large machines.

To minimize overall weight and size, miniature PM motors may use high energy magnets made with neodymium or other strategic elements; most such are neodymium-iron-boron alloy. With their higher flux density, electric machines with high-energy PMs are at least competitive with all optimally designed singly fed synchronous and induction electric machines. Miniature motors resemble the structure in the illustration, except that they have at least three rotor poles (to ensure starting, regardless of rotor position) and their outer housing is a steel tube that magnetically links the exteriors of the curved field magnets.

A series DC motor connects the armature and field windings in series with a common D.C. power source. The motor speed varies as a non-linear function of load torque and armature current; current is common to both the stator and rotor yielding current squared behavior[citation needed]. A series motor has very high starting torque and is
commonly used for starting high inertia loads, such as trains, elevators or hoists.

This speed/torque characteristic is useful in applications such as dragline excavators, where the digging tool moves rapidly when unloaded but slowly when carrying a heavy load. With no mechanical load on the series motor, the current is low, the counter-EMF produced by the field winding is weak, and so the armature must turn faster to produce sufficient counter-EMF to balance the supply voltage. The motor can be damaged by overspeed. This is called a runaway condition.

Series motors called "universal motors" can be used on alternating current. Since the armature voltage and the field direction reverse at the same time, torque continues to be produced in the same direction. Since the speed is not related to the line frequency, universal motors can develop higher-than-synchronous speeds, making them lighter than induction motors of the same rated mechanical output. This is a valuable characteristic for hand-held power tools. Universal motors for commercial utility are usually of small capacity, not more than about 1 kW output. However, much larger universal motors were used for electric locomotives, fed by special low-frequency traction power networks to avoid problems with commutation under heavy and varying loads.

A shunt DC motor connects the armature and field windings in parallel or shunt with a common D.C. power source. This type of motor has good speed regulation even as the load varies, but does not have the starting torque of a series DC motor.[3] It is typically used for industrial, adjustable speed applications, such as machine tools, winding/unwinding machines and tensioners.
A compound DC motor connects the armature and fields windings in a shunt and a series combination to give it characteristics of both a shunt and a series DC motor. This motor is used when both a high starting torque and good speed regulation is needed. The motor can be connected in two arrangements: cumulatively or differentially. Cumulative compound motors connect the series field to aid the shunt field, which provides higher starting torque but less speed regulation. Differential compound DC motors have good speed regulation and are typically operated at constant speed.

4.4.2 Induction Motor Model

The structural model represents CAN architecture aimed to ease communication between control station and server station connected to the process. The strategy assumes the control of the 3-phase induction motor is the scalar control using the Fuzzy and PI controllers which can be selected using a switch. Scalar control strategy is designed based on the steady state operation of an induction motor. Based on the mathematical equations the electrical dynamic of an induction motor in a synchronous rotating frame in the steady state, we obtain

\[ v_{ds} = r_s i_{ds} - \omega_s \Phi_{qs} \]  
(4.3)

\[ v_{qs} = r_s i_{qs} - \omega_s \Phi_{ds} \]  
(4.4)

\[ v_{dr} = r_r i_{dr} - (\omega_s - \omega) \Phi_{qr} \]  
(4.5)

\[ v_{qr} = r_r i_{qr} - (\omega_s - \omega) \Phi_{dr} \]  
(4.6)

The d and q axis can be referred in a space vector if they are placed at real and imaginary axis respectively. Hence,

\[ v_s = v_{ds} + jv_{qs} \]  
(4.7)
\[ i_s = i_{ds} + j i_{qs} \]  \hspace{1cm} (4.8)

\[ i_r = i_{dr} + j i_{qr} \]  \hspace{1cm} (4.9)

\[ \Phi_s = \Phi_{ds} + j \Phi_{qs} \]  \hspace{1cm} (4.10)

\[ \Phi_r = \Phi_{dr} + j \Phi_{qr} \]  \hspace{1cm} (4.11)

Deploying (4.3) and (4.4) in (4.7) yields:

\[ v_s = r_s i_s + j \omega_s \Phi_s \]  \hspace{1cm} (4.12)

Similarly, the rotor voltage of the induction motor is defined as:

\[ v_r = r_r i_r + j \omega_r \Phi_r \]  \hspace{1cm} (4.13)

The stator and rotor flux of the induction motor are:

\[ \Phi_{ds} = L_{si} q_s + M_{iqr} \]  \hspace{1cm} (4.14)

\[ \Phi_{qs} = L_{si} q_s + M_{iqr} \]  \hspace{1cm} (4.15)

\[ \Phi_{dr} = L_{rid} r + M_{ids} \]  \hspace{1cm} (4.16)

\[ \Phi_{qr} = L_{ri} q_r + M_{iqs} \]  \hspace{1cm} (4.17)

Based on the equation system and equations (4.9)-(4.12), the stator and rotor flux in vector space are defined by:

\[ \Phi_s = L_{sis} + M_{irs} \]  \hspace{1cm} (4.18)

\[ \Phi_r = L_{rir} + M_{irs} \]  \hspace{1cm} (4.19)

By replacing the flux with their expressions, (4.10) and (4.11) become:

\[ v_s = r_s i_s + j \omega_s [L_{si} i_s + M_{i_r}] \]  \hspace{1cm} (4.20)
The equivalent single phase model transformed to the stator where the magnetic leakages are added and grouped in the rotor and designed by Neówś[2]. Figure 4.5 shows the resulting model.

\[ 0 = r_L i_p + j \omega_p [L_p i_p + M i_e] \]  

(4.21)

Figure 4.5 Equivalent mono phase of the induction motor models

The scalar induction motor model is represented by a first order system. For the control of this system model, PI controller and Fuzzy controller are used. By fixing the desired closed loop dynamic model, we find the appropriate parameters of the PI controllers.

The scalar induction motor model is

\[ G_m(s) = \frac{k}{1+\tau s} \]  

(4.22)

where \( k = 0.3478 \) and \( \tau = 0.5184 \). \( \tau \) is the time constant and \( k \) is the static gain of the system.

\[ G(s) = \frac{F(s)}{G_m(s)(1-F(s))} \]  

(4.23)

\[ G(s) = k_p + \frac{1}{\tau_i(s)} \]  

(4.24)
Where \( k_p = 0.6 \) and \( \tau_i = 0.0899 \).

4.5 MODIFIED FUZZY PID CONTROLLER DESIGN

The use of a data network in a control loop has gained increasing attentions in recent years due to its cost effective and flexible applications. One of the major challenges in Networked Control System (NCS) is the network-induced delay effect in the control loop. The aim of the proposed Modified Fuzzy PID Logic Controller scheme is to improve the performance of the networked DC motor controller and also to compare the results with Fuzzy and Zeigler-Nichols tuned Networked Proportional-Integral-Derivative Controller. The performance of the proposed network controller has been simulated using MATLAB/SIMULINK and verified with real time systems.

Networked Control System is the adaptation of communication network for information exchange between controllers, sensors and actuators to realize a closed control loop. Networks reduce the complexity in wiring connections and the costs of Medias. They are easy to maintain and also enable remote data transfer and data exchanges among users. Because of these benefits, many industries and institutions have shown interest in applying different types of networks for their remote industrial control and automation.

In Recent Industrial control systems, Microprocessor based controllers are used to implement intelligent and reliable functions. In building such systems a grouping number of devices such as sensors and actuators are used. As the number of devices in a system groups, so data exchange in the system also increases. Various researches aim at making integrated control and communication algorithms to compensate for the unexpected change of network services. As control and feedback signals are transmitted in NCS, some amount of network delay is unavoidable because more than one station is sharing the network. A signal must be delayed if
there is another signal being transmitted on the network. This delay can be quite large and sometimes badly affect the performance of NCS.

Regardless of the types of networks, the overall performance of NCS is affected by two major challenges as networked induced delay and data losses. The challenges of networked DC motor are generally controlled by Conventional Proportional – Integral - Derivative Controllers, since they are less expensive with inexpensive maintenance, designed easily, and very effective. But mathematical model of the controller and tuning of PID parameters are difficult and generally not used for non-linear systems. Hence to overcome these challenges auto-tuning and adaptive PID Controller was developed with few mathematical calculations.

At first, a PID Controller is designed with a practical motor model. The transfer function model of the motor is then used to design the digital PID controller. The Fuzzy modeling is generated for self tuning of PID controllers. The above Fuzzy logic controller model is modified and simulated using MATLAB and Simulink by adding the network delays of NCS. The above results were verified on CAN bus connected PIC microcontroller network.

4.5.1 Modelling

The CAN bus connects the three nodes namely Sensor node, Controller node and Reference node. Each node consists of a microcontroller and a CAN BUS interface controller (Figure 4.6). Encoder pulse from speed sensor is used by Sensor node to calculate the speed of the motor and the information is transmitted to the controller node through CAN bus. The reference node gives the reference speed and the information is sent to controller node.
The controller node calculates the duty ratio and generates the PWM signal which is given to DC to DC converter [8].

![Diagram of control system](image)

**Figure 4.6 An overall real-time networked control system**

### 4.5.2 Fuzzy PID Controller

A proportional – integral – derivative controller (PID controller) is a generic control loop feedback mechanism (controller) widely used in industrial control systems – a PID is the most commonly used feedback controller [9]. A PID controller calculates an "error" value as the difference between a measured process variable and a desired set point. The controller attempts to minimize the error by adjusting the process control inputs.

The PID controller calculation (algorithm) involves three separate constant parameters, and is accordingly sometimes called three-term control: the proportional, the integral and derivative values, denoted P, I, and D. Heuristically, these values can be interpreted in terms of time: P depends on the present error, I on the accumulation of past errors, and D is a prediction of future errors, based on current rate of change. The weighted sum of these
three actions is used to adjust the process via a control element such as the position of a control valve, or the power supplied to a heating element.

In the absence of knowledge of the underlying process, a PID controller is the best controller. By tuning the three parameters in the PID controller algorithm, the controller can provide control action designed for specific process requirements. The response of the controller can be described in terms of the responsiveness of the controller to an error, the degree to which the controller overshoots the set point and the degree of system oscillation. Note that the use of the PID algorithm for control does not guarantee optimal control of the system or system stability.

Some applications may require using only one or two actions to provide the appropriate system control [10-12]. This is achieved by setting the other parameters to zero. A PID controller will be called a PI, PD, P or I controller in the absence of the respective control actions. PI controllers are fairly common, since derivative action is sensitive to measurement of noise, whereas the absence of an integral term may prevent the system from reaching its target value due to the control action.

The basic structure of the PID controller is first described in the following equations as well as Figure 4.7.

\[
U^C(t) = K_P + K_I \int edt + K_D \frac{de(t)}{dt}
\]  

(4.25)

\[
U^C(t) = K_P (1 + \frac{1}{T_I} \int dtd + T_D \frac{de(t)}{dt})
\]  

(4.26)
The two inputs-three outputs self tuning of a PID controller is used. The controller uses the error and change of error as inputs to the self tuning, and the gains (KP1, KI1, and KD1) as outputs. The FLC is adding to the conventional PID controller to adjust the parameters of the PID controller on-line according to the change of the signals error and change of the error. The controller also contain a scaling gains inputs (Ke, KΔe) as shown in Figure 4.8 to satisfy the operational ranges (the universe of discourse) making them more general.

Now the control action of the PID controller after self tuning can be describing as:

\[
U_c(t) = K_{p2} + K_{i2} \int e dt + K_{d2} \frac{d\varepsilon(t)}{dt}
\]

(4.27)

Where KP2, KI2, and KD2 are the new gains of PID controller and are equals to:

\[
KP2 = KP1 \times KP, \quad KI2 = KI1 \times KI, \quad \text{and} \quad KD2 = KD1 \times KD
\]

(4.28)
where KP1, KI1, and KD1 are the gains outputs of fuzzy control, that are varying online with the output of the system under control. And KP, KI, and KD are the initial values of the conventional PID controller.

![Figure 4.8 Fuzzy self tuning](image)

The general structure of fuzzy logic comprises of three principle components represented in Figure 4.9.

![Figure 4.9 Fuzzy general structure](image)

(i) **Fuzzification:**

This converts input data into suitable linguistic values. As shown in Figure 4.10 there are two inputs to the controller: error and rate change of the error signals.
The error is defined as:

$$e(t) = r(t) - y(t)$$  \hspace{1cm} (4.29)

Rate of error is defined as:

$$\Delta e(t) = \frac{de(t)}{dt}$$  \hspace{1cm} (4.30)

where $r(t)$ is the reference input, $y(t)$ is the output, $e(t)$ is the error signal, and $\Delta e(t)$ is the rate of error. The seventh triangular input and output membership functions of the fuzzy self tuning. For the system under study the universe of discourse for both $e(t)$ and $\Delta e(t)$ may be normalized from $[-1,1]$, and the linguistic labels are \{Negative Big, Negative, medium, Negative small, Zero, Positive small, Positive medium, Positive Big\}, and are referred to in the rules bases as \{NB, NM, NS, ZE, PS, PM, PB\}, and the linguistic labels of the outputs are \{Zero, Medium small, Small, Medium, Big, Medium big, very big\} and referred to in the rules bases as \{Z, MS, S, M, B, MB, VB\}.

![Figure 4.10 Membership function of inputs](image-url)
(ii) **Rule base**

A decision making logic which is simulating a human decision process inters fuzzy control action from the knowledge of the control rules and linguistic variable definitions.

Tables (4.1), (4.2), and (4.3) show the control rules that used for fuzzy self tuning of PID controller.

**Table 4.1 Rule bases for determining the gain KP1**

<table>
<thead>
<tr>
<th>$\dot{e}/e$</th>
<th>NB</th>
<th>NS</th>
<th>ZE</th>
<th>PS</th>
<th>PB</th>
</tr>
</thead>
<tbody>
<tr>
<td>NB</td>
<td>VB</td>
<td>VB</td>
<td>VB</td>
<td>VB</td>
<td>VB</td>
</tr>
<tr>
<td>NS</td>
<td>B</td>
<td>B</td>
<td>B</td>
<td>MB</td>
<td>MB</td>
</tr>
<tr>
<td>ZE</td>
<td>ZE</td>
<td>ZE</td>
<td>MS</td>
<td>S</td>
<td>S</td>
</tr>
<tr>
<td>PS</td>
<td>B</td>
<td>B</td>
<td>B</td>
<td>MB</td>
<td>VB</td>
</tr>
<tr>
<td>PB</td>
<td>VB</td>
<td>VB</td>
<td>VB</td>
<td>VB</td>
<td>VB</td>
</tr>
</tbody>
</table>
Table 4.2 Rule bases for determining the gain $K_{I1}$

<table>
<thead>
<tr>
<th>$\dot{e}/e$</th>
<th>NB</th>
<th>NS</th>
<th>ZE</th>
<th>PS</th>
<th>PB</th>
</tr>
</thead>
<tbody>
<tr>
<td>NB</td>
<td>M</td>
<td>M</td>
<td>M</td>
<td>M</td>
<td>M</td>
</tr>
<tr>
<td>NS</td>
<td>S</td>
<td>S</td>
<td>S</td>
<td>S</td>
<td>S</td>
</tr>
<tr>
<td>ZE</td>
<td>MS</td>
<td>MS</td>
<td>ZE</td>
<td>MS</td>
<td>MS</td>
</tr>
<tr>
<td>PS</td>
<td>S</td>
<td>S</td>
<td>S</td>
<td>S</td>
<td>S</td>
</tr>
<tr>
<td>PB</td>
<td>M</td>
<td>M</td>
<td>M</td>
<td>M</td>
<td>M</td>
</tr>
</tbody>
</table>

Table 4.3 Rule bases for determining the gain $K_{D1}$

<table>
<thead>
<tr>
<th>$\dot{e}/e$</th>
<th>NB</th>
<th>NS</th>
<th>ZE</th>
<th>PS</th>
<th>PB</th>
</tr>
</thead>
<tbody>
<tr>
<td>NB</td>
<td>ZE</td>
<td>S</td>
<td>M</td>
<td>MB</td>
<td>VB</td>
</tr>
<tr>
<td>NS</td>
<td>S</td>
<td>B</td>
<td>MB</td>
<td>VB</td>
<td>VB</td>
</tr>
<tr>
<td>ZE</td>
<td>M</td>
<td>MB</td>
<td>MB</td>
<td>VB</td>
<td>VB</td>
</tr>
<tr>
<td>PS</td>
<td>B</td>
<td>VB</td>
<td>VB</td>
<td>VB</td>
<td>VB</td>
</tr>
<tr>
<td>PB</td>
<td>VB</td>
<td>VB</td>
<td>VB</td>
<td>VB</td>
<td>VB</td>
</tr>
</tbody>
</table>

(iii) Defuzzification:

This yields a non fuzzy control action from inferred fuzzy control action. The most popular method, center of gravity or center of area is used for defuzzification. Where $u(u_j)$ membership grad of the element $u_j$, $u(nT)$ is the fuzzy control output, $n$ is the number of discrete values on the universe of discourse.

$$u(nT) = \frac{\sum_{j=1}^{n} u(u_j)u_j}{\sum_{j=1}^{n} u(u_j)}$$  \hspace{1cm} (4.31)
(iv) **Modified Fuzzy controller for NCS:**

The structure of FLC for a single input single output plant in a network is shown in Figure 4.12. In this figure, the control signal and plant output are transmitted through the network. Due to the use of the network, the control signal and feedback signal (plant output) inevitably contain the network induced delay and losses of data. In Figure 4.12 $r(t)$ is the reference input, $y(t)$ is the plant output, $e(t)$ is the error signal between the reference input and plant output and $U_c(t)$ is the control signal.

![Figure 4.12 Fuzzy logic controller for NCS](image)

**Figure 4.12 Fuzzy logic controller for NCS**

![Figure 4.13 Modified Fuzzy logic controllers for NCS](image)

**Figure 4.13 Modified Fuzzy logic controllers for NCS**
The modified fuzzy PID controller for the networked control dc motor is shown in the Figure 4.13. The model is based on modulating the control signal $U_c(t)$ provided by the Fuzzy PID controller with a single parameter $K$. The fuzzy modulator receives the input as the error signal $e(t)$ which is the difference between the reference signal and the plant output signal $y(t)$ in addition to the output from the fuzzy PID controller $U_C(t)$. The fuzzy modulator produces an output as modulation parameter $K$ which is used to compensate the affects of the network induced time delay and data losses.

The control signal produced by the fuzzy modulated networked PID controller is

$$U_F(t) = K \cdot U_C(t)$$  \hspace{1cm} (4.32)

Two fuzzy linguistic variables, i.e., Small and Large are defined. The coefficients of the membership functions are determined by several trial and error methods with the plant and without the network. The fuzzy logic modulator used in this paper is composed of the following rules.

If $e(t)$ is small and $U_{PID}(t)$ is small, then $K$ is $K_1$.

If $e(t)$ is large and $U_{PID}(t)$ is large, then $K$ is $K_2$

Such that $\beta < \beta_1 < \beta_2 < 1$ where $K_i$, $i=1,2$ are the consequent parameters corresponding to the modification parameter $K$.

The simulation scenario, the direct structure of the networked DC motor control system is simulated using MATLAB/ SIMULINK under fully controlled environments for Fuzzy Logic Controller, PID Controller and Modified Fuzzy PID Controller. The motor is controlled by the fuzzy logic controller with the insertions of network delays calculated for the CAN BUS. The delays are varied according to different effects of interests.
The disturbance and loss of input signal, control signal and the feedback signal were made for few milliseconds at each stage and the results were studied. Then the results of the FLC are compared with the Zeigler-Nichols tuned PID controller and fuzzy Modulated PID Controller.

Experimental verification for the above Modulated fuzzy PID controller in NCS (shown in Figure 4.14) has been performed to validate the simulation results. The DC motor is controlled digitally using PIC microcontrollers (PIC16F877) and CAN BUS interface controllers.
4.6 MODELLING OF CAN COMMUNICATION NETWORK

Figure 4.15 Networked Speed control system of DC Motor and Induction motor through CAN Bus

Figure 4.16 Equivalent mono phase of the induction Motor model
A CAN controller can be in one of three states:

- **Error active** - the normal operating mode for a controller. Messages can be received and transmitted. On detecting an error an active error flag is sent (see error signalling).

- **Error passive** - a mode entered when the controller has frequent problems transmitting or receiving messages. Messages can be received and transmitted. On detecting an error while receiving, a passive error flag is sent.

- **Bus off** - entered if the controller has serious problems with transmitting messages. No messages can be received or transmitted until the CAN controller is reset by the host microcontroller or processor.

The mode of the controller is controlled by two error counters - the transmit error counter (tx_count) and the receive error counter (rx_count). The following rules apply:

- The CAN controller is in error active mode if tx_count \( \leq 127 \) AND rx_count \( \leq 127 \).

- Passive mode is used if (tx_count > 127 OR rx_count > 127) AND tx_count \( \leq 255 \).

- Bus off is entered if tx_count > 255.

- Once the CAN controller has entered bus off state, it must be reset by the host microcontroller or processor in order to be able to continue operation. In addition, this is only allowed after the reception of 128 occurrences of 11 consecutive recessive bits.

- The counters are updated as follows:

- When a receiver detects an error, the rx_count will be increased by 1, except when the detected error was a bit error during the sending of an active error flag or an overload flag.
• When a receiver detects a dominant bit as the first bit after sending an error flag, the rx_count will be increased by 8.
• When a transmitter sends an error flag, the tx_count is increased by 8.
• Exception 1: If the transmitter is error passive and detects an ack error because of not detecting a dominant ack and does not detect a dominant bit while sending its passive error flag.
• Exception 2: If the transmitter sends an error flag because a stuff error occurred during arbitration whereby the stuff bit is located before the RTR bit, and should have been recessive, and has been sent as recessive but monitored as dominant.
• If a transmitter detects a bit error while sending an active error flag or an overload flag, the tx_count is increased by 8.
• If a receiver detects a bit error while sending an active error flag or an overload flag the rx_count is increased by 8.
• Any node accepts up to 7 consecutive dominant bits after sending an active or passive error flag or an overload flag. After detecting the 14th consecutive dominant bit (in the case of an active error flag or an overload flag), or after detecting the 8th consecutive dominant bit following a passive error flag, and after each sequence of additional 8 consecutive dominant bits every transmitter increases its tx_count by 8 and every receiver increases its rx_count by 8.

After the successful transmission of a message (getting ack and no error until end of frame is finished) tx_count is decreased by 1 unless it was already 0.

After the successful reception of a message (reception without error up to the ack slot and the successful sending of the ack bit), rx_count is
decreased by 1 if it was between 1 and 127. If rx_count was 0 it stays 0, and if it was greater than 127, it will be set to a value between 119 and 127.

Common tasks for interacting with a CAN network include:

- Connecting your PC to your CAN network using a variety of supported CAN interface hardware
- Sending and receiving messages using XCP protocol
- Filtering CAN messages
- Using CAN database files to encode and decode messages
- Generating code from CAN Simulink blocks

The design and implementation of armature-controlled and field-controlled DC motor system using both conventional PID and PID-like FLC have been presented. Comparisons of experimental results of the conventional PID controller and PID-like FLC show that the PID-like FLC is able to perform better than the conventional PID controller. Results indicate that even without knowing the detail of the control plants, we were able to construct a well performed fuzzy logic controller based on the experience about the position controller.

An overview of PID controller, design of PID controller using Z-N technique and design of fuzzy logic controller for higher order system. Simulation results using MATLAB / SIMULINK are discussed for Ziegler Nichols tuned PID controller, fine tuned PID controller and the Fuzzy logic controller. Ziegler Nichols technique gives high overshoot and settling time with zero steady state error.

Initial controller parameters obtained using Z-N formula need to be adjusted repeatedly through computer simulation to get satisfactory performance. Fine tuned PID controller gives zero steady state error and
smaller overshoot and settling time than Ziegler Nichols tuned PID controller. The Fuzzy Logic controller gives no overshoot, zero steady state error and smaller settling time than obtained using Ziegler Nichols tuned PID controller and fine tuned PID controller. The simulation results confirms that the proposed Fuzzy logic controller with simple design approach and smaller rule base can provide better performance comparing with the Ziegler Nichols tuned PID controller and fine tuned PID controller.

Once fuzzy relations are defined, it is possible to develop fuzzy relational databases. The first fuzzy relational database, FRDB, appeared in Maria Zemankova's dissertation. Later, some other models arose like the Buckles-Petry model, the Prade-Testemale Model, the Umano-Fukami model or the GEFRED model by J.M. Medina, M.A. Vila et al. In the context of fuzzy databases, some fuzzy querying languages have been defined, highlighting the SQLf by P. Bosc et al. and the FSQL by J. Galindo et al. These languages define some structures in order to include fuzzy aspects in the SQL statements, like fuzzy conditions, fuzzy comparators, fuzzy constants, fuzzy constraints, fuzzy thresholds, linguistic labels and so on.

Much progress has been made to take fuzzy logic database applications to the web and let the world easily use them, for This enables fuzzy logic matching to be incorporated into a database system or application. Fuzzy logic and probability address different forms of uncertainty. While both fuzzy logic and probability theory can represent degrees of certain kinds of subjective belief, fuzzy set theory uses the concept of fuzzy set membership, i.e., how much a variable is in a set (there is not necessarily any uncertainty about this degree), and probability theory uses the concept of subjective probability, how probable is it that a variable is in a set (it either entirely is or entirely is not in the set in reality, but there is uncertainty around whether it is or is not).
The technical consequence of this distinction is that fuzzy set theory relaxes the axioms of classical probability, which are themselves derived from adding uncertainty, but not degree, to the crisp true/false distinctions of classical Aristotelian logic.

Bruno de Finetti argues that only one kind of mathematical uncertainty, probability, is needed, and thus fuzzy logic is unnecessary. However, Bart Kosko shows in Fuzziness Probability that probability theory is a subtheory of fuzzy logic, as questions of degrees of belief in mutually-exclusive set membership in probability theory can be represented as certain cases of non-mutually-exclusive graded membership in fuzzy theory. In that context, he also derives Bayes' theorem from the concept of fuzzy subsethood. Lotfi A. Zadeh argues that fuzzy logic is different in character from probability, and is not a replacement for it. He fuzzified probability to fuzzy probability and also generalized it to possibility theory.

More generally, fuzzy logic is one of many different extensions to classical logic intended to deal with issues of uncertainty outside of the scope of classical logic, the inapplicability of probability theory in many domains, and the paradoxes of Dempster-Shafer theory. See also probabilistic logics.

The CFL (Compensatory Fuzzy Logic) is a branch of Fuzzy Logic. This is a new multivalent system that breaks with traditional axioms of such systems to achieve better semantic behavior to classical systems.

In processes involving decision making, trade with the experts leads to obtaining complex and subtle formulations and requires compound predicates. The truth values obtained on these compound predicates must possess sensitivity to changes in the truth values of basic predicates. This need is met by the use of the CFL, waiving compliance of the classical properties of conjunction and disjunction and rather opposing to them the idea that the
increase or decrease of the truth value of the conjunction or disjunction caused by change the truth value of one of its components can be compensated with a corresponding decrease or increase in the other. This increase or decrease in truth may be offset by the increase or decrease in another component. This notion makes the CFL logical and useful. There are cases in which compensation is not possible. This occurs when certain thresholds are violated and there is a veto preventing compensation.

Compensatory Fuzzy Logic consists of four continuous operators: conjunction (c), disjunction (d), fuzzy strict order (or) and negation (n). The conjunction is the geometric mean and its dual as conjunctive and disjunctive operators.

The modern automobile may have as many as 70 electronic control units (ECU) for various subsystems. Typically the biggest processor is the engine control unit. Others are used for transmission, airbags, antilock braking/ABS, cruise control, electric power steering, audio systems, power windows, doors, mirror adjustment, battery and recharging systems for hybrid/electric cars, etc. Some of these form independent subsystems, but communications among others are essential. A subsystem may need to control actuators or receive feedback from sensors. The CAN standard was devised to fill this need. The CAN bus protocol has been used on the Shimano Di2 electronic gear shift system.

Today the CAN bus is also used as a fieldbus in general automation environments, primarily due to the low cost of some CAN controllers and processors.
Manufacturers including NISMO aim to use CAN bus to recreate real-life racing laps in the videogame Gran Turismo using the game's GPS Data Logger function, which would then allow players to race against real laps.

The scalar induction motor model is:

\[ G_m(s) = \frac{k}{1 + \tau s} \]  

(4.33)

where, \( k = 0.3478 \) and \( \tau = 0.5184 \). \( \tau \) is the time constant and \( k \) is the static gain of the system:

\[ C(s) = \frac{F(s)}{G_m(s)(1 - F(s))} \]  

(4.34)

\[ C(s) = k_p + \frac{1}{\tau_i(s)} \]  

(4.35)

where, \( k_p = 0.6 \) and \( \tau_I = 0.0899 \).

In the simulation scenario, the direct structure of the networked DC motor control system is simulated using MATLAB/SIMULINK under fully controlled environments for Fuzzy Logic Controller, PID Controller and Self-adaptive fuzzy PID CAN based Controller for DC motor. The delays are varied according to different effects of interests. The disturbance and loss of input signal, control signal and the feedback signal were made for few milliseconds at each stage and the results are studied. The Figure 4.17 shows the comparison of the system performance for all DC motors with fuzzy, PID and fuzzy based PID in a network controlled environment without data losses.

Next in the study of Induction motor control system, simulated using MATLAB/SIMULINK under fully controlled environments for Fuzzy Logic Controller, PID Controller and Self-adaptive fuzzy PID CAN based
Controller for Induction motor. The delays are varied according to different effects of interests. The disturbance and loss of input signal, control signal and the feedback signal were made for few milliseconds at each stage and the results are studied.

Figure 4.17 (a)

(b)

Figure 4.17 (Continued)
Figure 4.17  (a, b and c) Fuzzy CAN based DC motor, Fuzzy PID CAN based DC motor and PID based DC motor

Figure 4.18 (Continued)
The equivalent single phase model transformed to the stator where the magnetic leakages are added and grouped in the rotor and designed by Neos. Figure 4.17 shows the resulting model. The scalar induction motor model is represented by a first order system. For the control of this system
model, PI controller and Fuzzy controller are used. By fixing the desired closed loop dynamic model, we find the appropriate parameters of the PI controllers.

The Figure 4.18 shows the comparison of the system performance for scalar control of the induction motor based on fuzzy, PI and fuzzy PI in the network controlled environment without data losses. Since the introduction of CAN in communication between control and server station and also fuzzy PID increases the throughput, it is observed that Fuzzy PID and CAN based Induction motor best suits this condition.

4.7 SUMMARY

Networks and their applications play a promising role for real-time high performance networked control in industrial applications. This paper presents Modified Fuzzy PID controller for CAN based networked control of DC motor. Simulations as well as experimental results are given to validate the algorithm. This presented model greatly improves the settling time and Maximum Peak overshoot. The analysis on using intelligent controls improves and strengthens the networked control systems concepts in the future.

Since fuzzy controllers are nonlinear, it is more difficult to set the controller gains compared to proportional-integral-derivative (PID) controllers. This design procedure and a tuning procedure that carries tuning rules from the PID domain over to fuzzy single-loop controllers. The idea is to start with a tuned, conventional PID controller, replace it with an equivalent linear fuzzy controller, make the fuzzy controller nonlinear, and eventually fine-tune the nonlinear fuzzy controller. This is relevant whenever a PID controller is possible or already implemented.
The development of self-adaptive fuzzy PID logic Networked Control of a DC motor and an Induction motor scalar drives using CAN Bus is discussed and developed. The comparative performance with various motor controls using CAN Bus is also prototyped. The aim of the proposed fuzzy PID Logic Controller scheme is to improve the performance of the networked DC motor and Induction motor controller and also to compare the results with Fuzzy, PID and Fuzzy based PID Networked Controller.

The performance of the proposed network controller has been simulated using MATLAB/SIMULINK. The results show self-adaptive fuzzy PID CAN based networked control suits on DC motor and self-adaptive fuzzy PI CAN based networked control suits for Induction motor.

When a traditional feedback control system is closed via shared communication network with other nodes outside the control system, the so-called systems can be classified as a Networked Control System (NCS). Control systems with spatially distributed components have existed for several decades. Examples include control systems in chemical process plants, refineries, power plants and airplanes. In traditional systems the nodes were connected via hardwired connections and the systems were designed to bring all the information from the sensors to a central location where the conditions were being monitored and decisions were made on how to control the system.

Networks and their applications play a promising role for real-time high performance networked control in industrial applications. This work presents Modified Fuzzy PID controller for CAN based networked control of DC motor. Simulations as well as experimental results are given to validate the algorithm. This presented model greatly improves the settling time and Maximum Peak overshoot.
After a period of time from operating the S.E.D.C. motor, by using feed forward controller, P, PI, PID and Fuzzy controller, where by comparing the results of the response of the output (speed) of the system (S.E.D.C. motor), it is evident that, feed forward, P, PI, and PID controllers that have been used for getting the desired design requirements for the system (S.E.D.C. motor) that some of desired results for system design requirements are not satisfied for improving the performance of the response of the output (speed) of the system (S.E.D.C. motor). After using Fuzzy controller all of desired design requirements for the system (S.E.D.C. motor) are satisfied for improving the performance of the response of output (speed) of the system (S.E.D.C. motor). Therefore fuzzy controller gives much more improved dynamic responses.

The analysis on using intelligent controls improves and strengthens the networked control systems concepts in the future. In the simulation scenario, the direct structure of the networked DC motor control system is simulated using MATLAB/SIMULINK under fully controlled environments for Fuzzy Logic Controller, PID Controller and Self-adaptive fuzzy PID CAN based Controller for DC motor. The delays are varied according to different effects of interests. The disturbance and loss of input signal, control signal and the feedback signal were made for few milliseconds at each stage and the results are studied. The comparison of the system performance for all DC motors with fuzzy, PID and fuzzy based PID in a network controlled environment without data losses.

Next in the study of Induction motor control system, simulated using MATLAB/SIMULINK under fully controlled environments for Fuzzy Logic Controller, PID Controller and Self-adaptive fuzzy PID CAN based Controller for Induction motor. The delays are varied according to different effects of interests. The disturbance and loss of input signal, control signal
and the feedback signal were made for few milliseconds at each stage and the results are studied. The comparison of the system performance for scalar control of the induction motor based on fuzzy, PI and fuzzy PI in the network controlled environment without data losses. Since the introduction of CAN in communication between control and server station and also fuzzy PID increases the throughput, it is observed that Fuzzy PID and CAN based Induction motor best suits for this condition.