CHAPTER 4
SOFT COMPUTING

This Chapter deals with the soft computing techniques used for the proposed work to optimize the controller for better control action and gives the fundamental idea to control the drives.

4.1 PARTICLE SWARM OPTIMIZATION

4.1.1 Introduction

Particle Swarm Optimization (PSO) is a technique used to explore the search space of a given problem to find the settings or parameters required to maximize a particular objective. This technique, first described by James Kennedy and Russell Eberhart in 1995, originates from two separate concepts: the idea of swarm intelligence based on the observation of swarming habits by certain kinds of animals (such as birds and fish); and the field of evolutionary computation.

PSO is one of the modern heuristic algorithms. It has been motivated by the behaviour of organisms, such as fish schooling and bird flocking. Generally, PSO is characterized as a simple concept, easy to implement, and computationally efficient. Unlike the other heuristic techniques, PSO has a flexible and well-balanced mechanism to enhance the global and local exploration abilities.

PSO is one of the optimization techniques and a kind of evolutionary computation technique. The method has been found to be robust in solving problems featuring non-linearity and non-differentiability, multiple
optima, and high dimensionality through adaptation, which is derived from
the social psychological theory. According to the research results for a flock
of birds, birds find food by flocking (not by each individual). The observation
leads the assumption that every information is shared inside flocking.
Moreover, according to observation of behavior of human groups, behavior of
each individual (agent) is also based on behavior patterns authorized by the
groups such as customs and other behavior patterns according to the
experiences by each individual.

4.1.2 PSO Computation

Optimization is the mechanism by which one finds the
maximum or minimum value of a function or process. This mechanism
is used in fields such as physics, chemistry, economics and engineering
where the goal is to maximize efficiency, production, or some other
measure. Optimization can refer to either minimization or maximization;
maximization of a function \( f \) is equivalent to minimization of the opposite
of this function (\( F \)). Mathematically a minimization task is defined as:

\[
f : \mathbb{R}^n \to \mathbb{R} \tag{4.1}
\]

Find \( x^\ast \in \mathbb{R}^n \) such that \( f (x^\ast) \leq f (x) \), \( \forall x \in \mathbb{R}^n \)

Similarly, maximization

Find \( x^\ast \in \mathbb{R}^n \) such that \( f (x^\ast) \geq f (x) \), \( \forall x \in \mathbb{R}^n \)

The domain \( \mathbb{R}^n \) of \( f \) is referred to as the search space (or
parameter space). Each element of \( \mathbb{R}^n \) is called a candidate solution in the
search space, with \( x^\ast \) being the optimal solution. The value \( n \) denotes the
number of dimensions of the search space, and thus the number of
parameters involved in the optimization problem. The function \( f \) is called the objective function, which maps the search space to the function space. Since a function has only one output, this function space is usually one-dimensional.

The function space is then mapped to the one-dimensional fitness space, providing a single fitness value for each set of parameters. This single fitness value determines the optimality of the set of parameters for the desired task. In most cases, the function space can be directly mapped to the fitness space. However, the distinction between function space and fitness space is important in cases such as multiobjective optimization tasks, which include several objective functions drawing input from the same parameter space.

### 4.1.3 PSO Algorithm

The PSO algorithm works by simultaneously candidate solutions in the search space. During each iteration of the algorithm, each candidate solution is evaluated by the objective function being optimized, determining the fitness of that solution. Each candidate solution can be thought of as a particle “flying” through the fitness landscape finding the maximum or minimum of the objective function. Initially, the PSO algorithm chooses candidate solutions randomly with in the search space. Figure 4.1 shows the initial state of a four-particle PSO algorithm seeking the global maximum in a one-dimensional search space. The search space is composed of all the possible solutions along the x-axis; the curve denotes the objective function.

Finally, the PSO algorithm maintains the best fitness value achieved among all particles in the swarm, called the global best fitness, and the candidate solution that achieved this fitness, called the global best position or global best candidate solution. The PSO algorithm consists of just
three steps, which are repeated until some stopping condition is met

1. Evaluate the fitness of each particle
2. Update individual and global best fitnesses and positions
3. Update velocity and position of each particle

Fitness evaluation is conducted by supplying the candidate solution to the objective function. Individual and global best fitnesses and positions are updated by comparing the newly evaluated fitnesses against the previous individual and global best fitnesses, and replacing the best fitnesses and positions particle in the swarm is updated using the following

\[ V_i(t + 1) = w v_i(t) + c1 r_1 [\hat{x}_i(t) - x_i(t)] + c2 r_2 [g(t) - x_i(t)] \]

(4.2)

The index of the particle is represented by \( i \).

Thus, \( v_i(t) \) is the velocity of particle \( i \) at time \( t \) and

\( x_i(t) \) is the position of particle \( i \) at time \( t \).

The parameters \( w, c_1, \) and \( c_2 \) (\( 0 \leq w \leq 1.2, 0 \leq c_1 \leq 2, \) and \( 0 \leq c_2 \leq 2 \)) are user-supplied coefficients. The values \( r_1 \) and \( r_2 \) (\( 0 \leq r_1 \leq 1 \) and \( 0 \leq r_2 \leq 1 \)) are random. Values regenerated for each velocity update. The value \( x^\ast i(t) \) is the individual best candidate solution for particle \( i \) at time \( t \), and \( g(t) \) is the swarm’s global best candidate solution at time \( t \).
Each of the three terms of the velocity update equation has different roles in the PSO algorithm. The first term $WVi(t)$ is the component, responsible for keeping the particle moving in the same direction it was originally heading. The value of the inertial coefficient twist typically between 0.8 and 1.2, which can either dampen the particle’s inertia or accelerate the particle in its original direction. Generally, lower values of the inertial coefficient speed up the convergence of the swarm to optima, and higher values of the inertial coefficient encourage exploration of the entire search space of the global best candidate solution, or a predefined target fitness value.

4.2 GENETIC ALGORITHM

Genetic algorithm (GA) uses the principles of evolution, natural selection and genetics from natural biological systems in a computer algorithm to simulate evolution. Essentially, the genetic algorithm is an optimization technique that performs a parallel, stochastic, but directed search to evolve the fittest population. The idea, in all the system based on Genetic
algorithm, was to evolve a population of candidate solutions to a given problem, using operators inspired by natural genetic variation and natural selection.

**Reproduction**

By using the values of the performance fitness functions, select the best N/2 individuals of the current generation to be the become parents for producing the Next generation. This means that only genetically good individuals are selected to parent.

**Crossover**

Two parents are randomly selected to exchange the genetic information with each other and two new individuals are generated so as to keep the population size at constant value N.

**Mutation**

Mutation plays a secondary role in genetic algorithms. It is needed because, occasionally, chromosomes may lose some potentially useful genetic material. Mutation takes place with a certain probability; thus genetic content of a particular individual gets changed and a new generation is produced. Mutation is important in nature as it brings a change in genetic content of the individuals in order to enable them to adapt to a different environment. In the same way, in artificial systems the mutation will direct the search algorithm to a new search space so that a global minima can be found.

**Fitness function**

A fitness function takes a chromosome as an input and returns a number that is a measure of the chromosome’s performance on the problem to be solved. Fitness function plays the same role in GA as the environment plays in natural evolution. The interaction of an individual with its
environment provides a measure of fitness to reproduce. Similarly the interaction of a chromosome with a fitness function provides a measure of fitness that the GA uses while carrying out reproduction. Genetic algorithm is a maximization routine; the fitness function must be a non-negative Figure of merit. In this particular situation our main aim is to minimize error and reduce the rise time and overshoot. Hence the fitness function, in this case, is a function of error and rise time.

\[
J = \int (W_1 |e(t)| + W_2 u^2(t) + W_3 t_r)
\]  \hspace{1cm} (4.4)

where, \( w_1, w_2, w_3 \), are the weight coefficients \( u(t) \) is the output of the controller \( e(t) \) is the error.

The square term of control output is added to overcome the large energy of the controller. The fitness function is chosen as,

\[
f = \frac{1}{J + 10^{-8}}
\]  \hspace{1cm} (4.5)

The term \( 10^{-8} \) is added in the denominator of fitness function to avoid it from becoming zero.

**Fitness proportionate selection with Roulette Wheel**

The original GA used fitness proportionate selection, in which the "expected value" of an individual (i.e., the expected number of times an individual will be selected to reproduce) is that individual's fitness divided by the average fitness of the population. The most common method for implementing is "roulette wheel" sampling. Each individual is assigned a slice
of a circular "roulette wheel", the size of the slice being proportional to the individual's fitness. The wheel runs \( N \) number of times, where \( N \) is the number of individuals in the population. On each spin, the individual under the wheel's marker is selected to be in the pool of parents for the next generation.

### 4.3 ANT COLONY SEARCH ALGORITHM (ACSA)

The Ant Colony Search Algorithms (ACSA) are especially suited for finding solutions to different optimization problems. A colony of artificial ants cooperates to find good solutions, which are an emergent property of the ant’s cooperative interaction. Based on their similarities with ant colonies in nature, ant algorithms are adaptive and robust and can be applied to different versions of the same problem as well as to different optimization problems.

The ACSA algorithm is used to optimize the gains and the values are applied into the controller of the plant. The objective of this algorithm is to optimize the gains of the PID controller for the given plant. The proportional gain makes the controller respond to the error while the integral derivative gain help to eliminate steady state error and prevent overshoot respectively.

The communication between the ants is mediated by the deposition of pheromone to the elements of good solutions. Then the elements with a higher quantity of pheromone become more attractive for the other ants. The quantity of pheromone deposited on each element is a function of the quality of the solution.
The Algorithm for Ant colony optimization is given step by step procedure

**Step I** : Initialize randomly potential solutions of the parameters $K_p$, $K_i$, $K_d$ by using uniform distribution. Initialize the pheromone trail and the heuristic value

**Step II** : Place the $A^{th}$ ant on the node. Compute the heuristic value associated on the objective (minimize the error)

**Step III** : Use pheromone evaporation equation to avoid unlimited increase of pheromone trails and allow the forgetfulness of bad choices

**Step IV** : Evaluate the obtained solutions according to the objectives

**Step V** : Display the optimum values of the optimization parameters

**Step VI** : Update the pheromone, according to the optimum solutions calculated at

**Step VII** : Iterate from step II until the maximum of iterations is reached

By using biological inspired ACSA technique it is found that the closed loop system has very fast rise time, settling time and zero maximum overshoot to sustain the system stability under servo condition. From the transient response, it is observed that ACSA method gives fast response. The ACSA method is better as compared to conventional PID controller. It is also found that the modulus margin does not fall in the range which is required for the same system. Also comparing with conventional PID tuning rules and optimal tuning of PID controller using error integrals, the ACSA technique has the advantages of system being faster rise time, settling time and zero maximum overshoot.
4.4 SOFT COMPUTING BASED CONTROLLER DRIVES

4.4.1 Block Diagram of BLDC Drive

Two controlling loops are used to control BLDC motor. The inner loop synchronizes the inverter gate signals with the electromotive forces. The outer loop controls the motor’s speed by varying the dc bus voltage. By using hall sensor information and sign of reference current, the decoder block generates signal vector of Back EMF is shown in Figure 4.2. Based on the signals from the controlling loops the switching signals are generated and controlling of the motor is done.

![Block Diagram of BLDC Drive](image)

**Figure 4.2 Block Diagram of BLDC Drive**

The Hall-effect was discovered in year 1879 by Edwin Hall and explains what is happening when a magnetic field is applied orthogonally to a metal plate. If a constant current flows through a piece of metal, the perpendicular voltage over the same metal will change with the variation in
strength of the applied magnetic field. Equation shows how the Hall-voltage varies

\[ U_H = \frac{IB}{qN_d} \]  

(4.6)

where I is the current in Ampere, B is the applied magnet field in Tesla, q is the charge of an electron in Coulombs, N is the carrier density in carriers/cm³ and d is the thickness of the conductor in metre.

This effect can be used to sense variations in magnetic fields, e.g. from a rotating magnet. One way to use this sensor in a PMSM is to mount three sensors together with filtering and amplification on one electrical turn of the machine and let those sense the magnetic field of the permanent magnets. This waveforms varies for different operating points of the motor during motor running.

4.4.2 Position Sensor

BLDC motor technology, coupled with SS Series Hall-effect sensors, provides a cost effective and reliable alternative to conventional DC motor commutation. Manufacturing and operating cost savings can be realized in several areas. Position sensors are relatively low cost devices. The sensors’ low operate and release Gauss levels allow the manufacturer to install lower-cost commutation magnetics. These solid-state sensors have no moving parts to wear out, eliminating maintenance and performance problems created by brush wear and the associated dust.

Brushless DC motors differ from brush type DC motors in that they employ electronic (rather than mechanical) commutation of the windings. Figure 4.3 shows how this electronic commutation can be performed by three digital output bipolar or bipolar latching sensors. Permanent magnets mounted on the rotor shaft operate the sensors. The sensors communicate the angular
position of the shaft to a logic circuit, which encodes this information and controls switches in a driver circuit. The windings then alternate in polarity, in effect rotating in relation to the shaft position. The windings react with the field of the rotor’s permanent magnets to develop the required torque.

![Figure 4.3 BLDC Sensors](image)

### 4.4.3 Block Diagram of Switched Reluctance Motor (SRM) Drive

In this proposal a PID controller is suggested for the speed control of the SR Motor shown in Figure 4.4 and the proposed PID controller parameters are optimized by using Genetic Algorithm (GA). GA is one of the most efficient methods in soft computing techniques. By using this method the speed control and also the problems of SR Motor such as torque ripples and acoustic noises are reduced this is done by changing the switching ON, OFF timings.

![Figure 4.4 Block Diagram of SRM Drive](image)
4.4.4 **Block Diagram of Permanent Magnet Synchronous Motor (PMSM) Drive**

PMSM motor control models were mathematically extracted and implemented using Fuzzy PID and Neuro Fuzzy PID Controller. To overcome the maximum overshoot, artificial intelligence techniques have been incorporated in the controller architecture. Fuzzy logic controlled model of DC motor is implemented, investigated and further optimized by the soft computing techniques for the optimal fuzzy rule base.

PID Controller is one of the most widely used controllers for the purpose of controlling electrical drives. The controller takes a measured value from a process or other apparatus and compares it with reference set point values. PID controller's algorithm are mostly used in feedback loops. PID controllers can be implemented in many forms. It can be implemented as a stand-alone controller or as part of Direct Digital Control (DDC) package or even Distributed Control System (DCS) is shown in Figure 4.5. The latter is a hierarchical distributed process control system which is widely used in process plants such as iron and steel or oil refining industries.

![Figure 4.5 Block Diagram of PMSM Drive](image-url)
Greater change is greater response, good for dampening spikes and jumps. The controller takes a measured value from a process or other apparatus and compares it with reference set point values.

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. The rules are

i) All the rules that apply are invoked, using the membership functions and truth values obtained from the inputs, to determine the result of the rule.

ii) This result in turn will be mapped into a membership function and truth value controlling the output variable.

iii) These results are combined to give a specific ("crisp") answer, the actual brake pressure, a procedure known as "defuzzification".

This combination of fuzzy operations and rule-based "inference" describes a "fuzzy expert system".