A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF ÇANKIRI KARATEKİN UNIVERSITY

PID PARAMETERS TUNING BASED ON DANDELION OPTIMIZATION ALGORITHM FOR DC MOTOR

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER OF SCIENCE
IN
ELECTRICAL AND ELECTRONICS ENGINEERING

BY

KHALAF ABDULLAH KHALAF KHALAF

ÇANKIRI

2023

PID PARAMETERS TUNING BASED ON DANDELION OPTIMIZATION ALGORITHM FOR DC MOTOR

By Khalaf Abdullah Khalaf KHALAF

August 2023

We certify that we have read this thesis and that in our opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science

Advisor **:** Asst. Prof. Dr. Mustafa TEKE

Examining Committee Members:

Chairman **:** Asst. Prof. Dr. Zafer CİVELEK
Electrical and Electronics Engineering
Çankırı Karatekin University

Member **:** Asst. Prof. Dr. Fuat TÜRK
Computer Engineering
Kırıkkale University

Member **:** Asst. Prof. Dr. Mustafa TEKE
Electrical and Electronics Engineering
Çankırı Karatekin University

Approved for the Graduate School of Natural and Applied Sciences

Prof. Dr. Hamit ALYAR
Director of Graduate School

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Khalaf Abdullah Khalaf KHALAF

ABSTRACT

PID PARAMETERS TUNING BASED ON DANDELION OPTIMIZATION ALGORITHM FOR DC MOTOR

Khalaf Abdullah Khalaf KHALAF
Master of Science in Electrical and Electronics Engineering
Advisor: Asst. Prof. Dr. Mustafa TEKE
August 2023

In various applications demanding precise control over position and speed, such as robotics and CNC machines, the utilization of direct current (DC) motors has been widely acknowledged. In this context, an exhaustive investigation into a permanent magnet DC motor (PMDC) has been undertaken in this thesis. A cascaded P-PI controller has been formulated and executed to manage the rotational speed and angular position of the motor. Comprised of three individual controllers, the cascaded configuration has been engineered such that the first controller is committed to current regulation, the second to speed adjustment, and the third to position governance. To optimize the control methodology for the DC motor, various algorithms have been explored. The Classic Method (CM), genetic algorithm (GA), butterfly optimization algorithm (BOA), and dandelion optimization algorithm (DOA) have been applied to this purpose. Performance criteria have been employed to compare these four algorithms. It was found that DOA exhibited the shortest rise time of 0.076 seconds and the least overshoot of 3.13%, thus being identified as the superior option for motor control. Following closely, BOA demonstrated a rise time of 0.08 seconds and a somewhat elevated overshoot of 4%. On the other hand, GA was observed to be less efficacious with a protracted rise time of 0.35 seconds, settling time of 0.5 seconds, and the highest overshoot at 8.1128%. Optimization of the PID controller parameters was subsequently carried out using DOA, in light of its exceptional response to time-domain features and its attainment of desired performance metrics. The cascaded P-PI controller, when optimized with DOA, has been shown to facilitate superior control over the PMDC motor's speed and position. By virtue of this optimization, an enhanced level of accuracy and stability has been achieved, making the system more reliable and efficient in practical applications that necessitate rigorous control over motor variables. Through systematic evaluation, it has been revealed that the cascaded P-PI controller's architecture enables a more refined and effective control strategy. Additionally, the benefit of employing optimization algorithms for fine-tuning the controller has been distinctly verified. Among the algorithms scrutinized, DOA emerged as the most effective in optimizing the PID controller parameters, thereby substantiating its applicability and efficacy in real-world scenarios. The findings of this thesis have significant implications for industries and applications where precise motor control is paramount. Through the implementation of the cascaded P-PI controller optimized by DOA, an advancement in the state of the art in motor control has been realized. The research has successfully addressed the inherent challenges related to speed and position control in PMDC motors and has laid down a robust foundation for future work in this domain. Thus, a comprehensive understanding of the comparative advantages and limitations of various optimization algorithms in the context of PMDC motor control has been achieved, culminating in the identification of DOA as the preeminent choice for optimization.

2023, 63 pages

Keywords: Permanent magnet DC motor, Dandelion optimizer algorithm, PID controllers

ÖZET

DC MOTOR İÇİN DANDELION OPTİMİZASYON ALGORİTMASINA DAYALI PID PARAMETRELERİ AYARI

Khalaf Abdullah Khalaf KHALAF
Elektrik ve Elektronik Mühendisliği, Yüksek Lisans
Tez Danışmanı: Dr. Öğr. Üyesi Mustafa TEKE
Ağustos 2023

Konum ve hız üzerinde kesin kontrol gerektiren çeşitli uygulamalarda, örneğin robotik ve CNC makinelerinde, doğru akım (DC) motorların kullanımı geniş bir şekilde kabul görmüştür. Bu bağlamda, bu tezde sürekli mıknatıslı bir DC motor (PMDC) üzerine kapsamlı bir inceleme yapılmıştır. Motorun döner hızı ve açısal konumu yönetmek için kademeli bir P-PI denetleyici formüle edilmiş ve uygulanmıştır. Üç ayrı denetleyiciden oluşan kademeli yapı, ilk denetleyicinin akım regülasyonuna, ikincisinin hız ayarlamasına ve üçüncüsünün konum yönetimine adanmış şekilde tasarlanmıştır. DC motoru için kontrol metodolojisini optimize etmek amacıyla çeşitli algoritmalar keşfedilmiştir. Klasik metod (CM), genetik algoritma (GA), kelebek optimizasyon algoritması (BOA) ve karahindiba optimizasyon algoritması (DOA) bu amaç için uygulanmıştır. Bu dört algoritmayı karşılaştırmak için performans kriterleri kullanılmıştır. DOA'nın en kısa yükselme süresini 0.076 saniye ve en az aşımı %3.13 ile gösterdiği, dolayısıyla motor kontrolü için üstün bir seçenek olduğu belirlenmiştir. Hemen ardından BOA, 0.08 saniye yükselme süresi ve biraz daha yüksek aşım oranı olan %4 ile gösterilmiştir. Öte yandan, GA'nın daha uzun bir yükselme süresi olan 0.35 saniye, yerleşme süresi olan 0.5 saniye ve en yüksek aşım oranı olan %8.1128 ile daha az etkili olduğu gözlemlenmiştir. PID denetleyici parametrelerinin optimizasyonu, zaman etki alanına olan olağanüstü yanıtı ve arzu edilen performans ölçütlerine ulaşması nedeniyle DOA kullanılarak sonrasında gerçekleştirilmiştir. DOA ile optimize edildiğinde, kademeli P-PI denetleyicinin PMDC motorunun hızı ve konumu üzerinde üstün kontrol sağladığı gösterilmiştir. Bu optimizasyon sayesinde, daha yüksek bir doğruluk ve kararlılık seviyesi elde edilmiş, böylece motor değişkenleri üzerinde titiz kontrol gerektiren pratik uygulamalarda sistem daha güvenilir ve verimli hale getirilmiştir. Sistematik bir değerlendirme aracılığıyla, kademeli P-PI denetleyicinin mimarisi daha rafine ve etkili bir kontrol stratejisi sağlamakta olduğu ortaya çıkmıştır. Ayrıca, denetleyiciyi ince ayarlamak için optimizasyon algoritmalarını kullanmanın faydası açıkça doğrulanmıştır. İncelenen algoritmalar arasında, DOA PID denetleyici parametrelerini optimize etmede en etkili olanı olarak ortaya çıkmış, böylece gerçek dünya senaryolarında uygulanabilirliği ve etkinliği doğrulanmıştır. Bu tezin bulguları, kesin motor kontrolünün hayati önem taşıdığı endüstriler ve uygulamalar için önemli sonuçlar taşımaktadır. DOA tarafından optimize edilen kademeli P-PI denetleyicinin uygulanmasıyla, motor kontrolündeki sanatın durumu ilerlemiştir. Araştırma, PMDC motorlarındaki hız ve konum kontrolü ile ilgili mevcut zorlukları başarıyla ele almış ve bu alandaki gelecekteki çalışmalar için sağlam bir temel atmıştır. Böylece, PMDC motor kontrolü bağlamında çeşitli optimizasyon algoritmalarının karşılaştırmalı avantajları ve sınırlılıkları hakkında kapsamlı bir anlayış elde edilmiş, DOA'nın optimizasyon için önde gelen seçenek olarak tanımlanmasına neden olmuştur.

2023, 63 sayfa

Anahtar Kelimeler: Kalıcı mıknatıslı DC motor, Dandelion optimizer algoritması, PID kontrolörleri

PREFACE AND ACKNOWLEDGEMENTS

I would like to thank my thesis advisor, Asst. Prof. Dr. Mustafa TEKE, for his patience, guidance and understanding.

Khalaf Abdullah Khalaf KHALAF
Çankırı-2023

CONTENTS

LIST OF SYMBOLS

c_1 ,c_2	Acceleration coefficients
θ	Actual position (rad)
L_a	Armature inductance (H)
R_a	Armature resistance (Ω)
V_{AN}, V_{BN}	Average output of converter (v)
K_v	Back emf constant (v sec/rad)
K_c ,K_s	Coupling and synchronization factors
ωci, fci	Crossover and switching frequency (H)
I	Current (A)
τ_e	Electrical time constant (sec)
T_m	Electromagnetic torque (Nm)
E_a	Electromotive force or back emf (V)
B_m	Friction coefficient (Nm.sec /rad)
$KPWM$	Gain of PWM
K_i	Integral gain
L	Length of conductor (m)
B	Magnetic field (weber/m^2)
F	Mechanical force N
τ_m	Mechanical time constant (sec)
J_m	Moment of inertia (kg.m^2)
ω_m	Motor speed (rad/sec)
T_L	Nominal load torque (Nm)
Va	Nominal voltage (v)
a	Power exponent
ρ	Probability switch
K_P	Proportional gain
r_1 ,r_2	Random coefficients
c	Sensory modality
$\mathcal{E}e$	Synchronization error
K_t	Torque constant (Nm/A)
V_{tri}	Triangular voltage (v)
Vc	Voltage control (v)
W	Weighted inertia

LIST OF ABBREVIATIONS

AC	Alternating current
ACO	Ant colony optimization
BBO	Biogeography based optimization algorithm
BH	Black hole
BOA	Butterfly optimization algorithm
CM	Classical method
CBO	Colliding bodies optimization
CNC	Computer numerical control
CCT	Cross coupling technique
DOA	Dandelion optimizer algorithm
DE	Differential evolution
DC	Direct current
ES	Evolution strategy
GA	Genetic algorithm
IAE	Integral absolute error
ISE	Integral square error
ITAE	Integral time absolute error
ITSE	Integral time square error
LB	Lower bound
MMF	Magnetic motive force
PSO	Particle swarm optimization
PMDC	Permanent magnet DC motor
PD	Proportional derivative
PID	Proportional integral derivative
P-PI	Proportional proportional integral
PSSA	Proposed simple synchronization algorithm
PWM	Pulse width modulation
Tr	Rising time
SMC	Sliding mode controller
Ts	Settling time
UB	Upper bound

LIST OF FIGURES

LIST OF TABLES

1. INTRODUCTION

An electric motor is fundamentally an electrical machine that transforms electrical energy into kinetic (mechanical) energy (Bigelow 2020, Franchi 2019). Motors are broadly categorized into DC (direct current) and AC (alternating current) types, based on their power supply sources. DC motors can be powered by direct current sources like batteries, solar panels, and more. On the other hand, AC motors rely on alternating current sources such as generators and power grids (Kang 2016). It's worth noting that DC motors can be powered by AC supply sources, provided the AC is converted to DC using rectifier circuits. Similarly, AC motors can be operated using direct current sources if the DC is converted to AC using an inverter.

Momoh (2018), Krause *et al.* (2002) utilized a specific type of DC motor, the permanent magnet motor (PMDC). Employed across various applications, the PMDC shares structural and operational similarities with the shunt-connected DC motor To control the primary functions of the PMDC, specifically its velocity and position, we implemented a cascade control system. This system, comprised of three controllers, regulates the current, speed, and position of the motor (Son *et al.* 2014, Cankurtaran and Kocamis 2019). Such controllers not only provide the desired system response and minimize errors, but they also ensure the system quickly returns to a steady state under a specific load. The primary advantage of using these controllers over others is their capacity to regulate multiple motor functions, such as speed, current, and position, which would be challenging to manage manually (Wang 2020, Raja and Ali 2017). As noted by (Cuong and Puta 2013), the model evolved to incorporate dual motors rather than a single DC motor, enhancing their utility across various applications, especially when distributing specific loads. This model finds its use in a myriad of applications. Some, like paper and textile mills, demand precise speed control, while others, such as metal cutting machines, CNC machines, and robotics, require meticulous and accurate position control (Tang 2001, Romero and Concha 2006).

This work was conducted using the Matlab environment and implemented in real-time. The results demonstrated high accuracy in achieving the specified position at a regular speed in both the presence and absence of load conditions. The simulation results using the Matlab environment displayed high precision in reaching the specified position of the PMDC motor, with precise trajectory tracking, complemented by the accurate control of the motor's movement along the axis.

1.1 Problem Statement

Researchers, especially those in the area of robotics, have recently shown an increased interest in the difficulty of regulating DC motor speed and positioning. How to precisely coordinate the motion of these motors along various axes is also a major focus of research. Below is a brief description of the issues addressed in this study.

1) How can we determine the values of these controllers' parameters in a way that guarantees accurate outcomes and quick steady-state recovery following a temporary load reduction?
2) Maintain desired position and speed despite changes in load, voltage or other disturbances.
3) Model the dynamic behavior of the DC motor accurately.
4) Design a suitable control strategy that is robust and stable.
5) Achieve precise and efficient motor operation in a wide range of operating conditions.

Also, Optimization algorithm is used to find the best set of control parameters for the control system to achieve optimal performance. It can search through a large parameter space to find the optimal values for the control system, which can be difficult to do manually. Optimization algorithms can improve the accuracy and efficiency of the control system. They can also account for non-linear behavior in the motor, which is difficult to model and control using traditional methods. Optimization algorithms can be used to achieve a range of performance objectives, such as minimizing overshoot, reducing settling time, or maximizing system stability.

1.2 Thesis Objective

The primary objective of this study can be divided into two parts. The first part focuses on utilizing a single PMDC motor and controlling its main functions, namely speed and position. The second part aims to integrate novel optimization algorithms with the PMDC motor's modeling to achieve robust performance. Below is a summary of the specific goals for this study:

1) Using Matlab, design a cascade control system to regulate the rotational speed and angular position of the PMDC motor.
2) For extracting the optimal parameter values for the P and PI controllers within the cascade control system the Genetic Algorithm (GA), Butterfly Optimization Algorithm (BOA), and Dandelion Optimizer Algorithm (DOA) are three cutting-edge optimization techniques that will be employed to achieve this goal. The aim is to select the best parameters that yield the most accurate results and rapidly return to a steady-state when a specific load is applied. These optimization algorithms will be compared to ascertain the most effective one.
3) Minimizing the error rate between the input and output to the lowest possible level to ensure the precision of the results. This will be achieved using multiple performance criteria functions, also known as fitness functions.
4) Construct a model compatible with optimization algorithms for the speed and position control of the PMDC motor.

The primary objective of this study is to design and implement a speed and position control circuit for a single PMDC motor. The aim is to ensure accurate tracking along the movement path and achieve the desired position with high precision. Following this, the model was augmented by integrating optimization algorithms like GA, BOA, and DOA. This integration facilitates highly accurate synchronized control over both position and speed. To validate the results' accuracy, we employed four methods to derive the parameter values for the P and PI controllers: Classical Control Method (CM), GA, BOA, and DOA. Subsequent to these applications, a comparison was conducted to ascertain the method that delivered the most optimal values..

1.3 Thesis Layout

This study contain of six chapters. The first chapter introduces the subject, outlines the problem statement, and presents the research methodology. The second chapter provides a literature review of previous works. The third chapter delves into the modeling and optimal control theory of a PMDC motor. This includes an introduction to the PMDC motor, its applications, and its advantages and disadvantages. Additionally, it discusses the type of controller used and the optimal methods for determining the P and PI parameter values. The chapter concludes by exploring the performance indices used in this thesis. Chapter 4 delves into the position and speed control of PMDC motors when they operate in tandem. It covers the modeling of this system, the methods to precisely synchronize the motors' movement along the axis, and the applications of this system. Chapter five presents the results and discussions of this study. The final chapter concludes the study and suggests potential avenues for future research in this domain.

2. LITERATURE REVIEW

PID controllers are well-liked because of its many desirable characteristics, including their ease of use, openness, dependability, efficiency, and durability. Parameter tuning is an area of study that seeks to reduce unwanted fluctuations and maximize response speed. In this paper, we take a look back at some of the traditional and cutting-edge methods for tweaking the PID controller's settings with the use of metaheuristics. It classifies approaches as either classical or metaheuristic, provides information about metaheuristic algorithms, and identifies areas where more study is needed.

Yeh and Hsu (2000) present a system for speed and position control of a DC motor designed to function within a closed-loop system. This system comprises an Actuator DC motor, a sensor to measure position, and the mechanical load that the motor is tasked to move. The control algorithm for this setup has been implemented using Simulink and embedded code.

Tang (2001) presents a PID controller design aimed at controlling the speed and position of a DC motor utilizing a cost-effective digital signal processor, specifically the TMS320C31 suite. Moreover, real-time values for the speed and position controller parameters are adjusted directly and online during the DC motor's operational period, ensuring continuous and uninterrupted operation.

Similarly, another study employed a cascade PID control to regulate the speed and position of a non-rigid joint. This model features a DC motor linked to a non-rigid joint, which includes a damper, four springs, and a gearbox. The gearbox serves as the connecting tool between the motor and the joint. For the gearbox parameters, the researcher utilized an estimation method (Sun and Mills 2002).

Ciucur (2004) presents a design for controlling the speed and position of a DC motor using fractional-order PI and PD controllers. These controllers are designed based on time domain specifications. A comparative analysis is conducted between the fractional-order

PI and PD controllers and their integer-order counterparts, demonstrating the superior performance of the former.

Pisano *et al.* (2008) presents a system for controlling the speed and position of a moving robot using a brushless DC motor, wherein a motor is connected to each wheel. The three-phase bridge is designed using N-mosfet and an electronic circuit to drive the motors. PWM, the outputs of both the optical encoder, and the Hall sensor are used to implement closed-loop control of speed and position.

Mhase *et al.* (2012) presents an integrated movement system that combines feedforward and feedback with multi-axis cross-coupled control (CCC) in a CNC machine to achieve superior path tracking performance. To design this controller efficiently, a complete system analysis and a new formulation of the contouring error transfer function (CETF) for multi-axis systems are performed.

Chen *et al.* (2013) combines cross-coupling technology with an adaptive control architecture to synchronize multiple robots. The success of this technology lies in ensuring that each robot tracks its desired position while synchronizing its movement with the rest of the robots, thereby ensuring position errors between robots converge to zero.

Xiao *et al.* (2013) develops a robust motion control strategy for dual DC drive systems by combining second-order sliding mode control techniques. Firstly, a speed control unit is designed for one motor to eliminate torque disturbances. Secondly, a control unit is designed using the cross-coupled control (CCC) method to reduce speed error and the unbalanced torque of both motors.

Ishizaki *et al.* (2013) uses a novel master-slave control method to synchronize the movement of two-drive systems. This method combines reference adaptive control with variable structure control to reduce synchronous motion error in real-time. The experiments were conducted on a gantry-type two-axis platform, moving a 5kg metal piece along the x-axis to a specified position.

Gücin *et al.* (2015), presents the design and implementation of a bi-directional DC motor speed and position control system using ATMEGA32 microcontrollers and LabVIEW software. An optical encoder is connected to the motor shaft to provide feedback on speed and position angle. The motor is driven using the PWM technique, while the LabVIEW software uses the graphical user interface to input the reference signal, whether speed or position angle.

To regulate the rotational speed and angular position of the PMDC, (Taut *et al.* 2015) developed a cascade PI(D) controller. It employs three methods to adjust the PID gains: a classic method and two optimal methods. These methods are compared to obtain the best results for speed and position.

In domains demanding precise regulation, PID controllers are paramount. The tuning of their parameters has long been a subject of study due to the profound impact on performance. While traditional methods for tuning often struggle with non-linearities and non-minimum phase characteristics, contemporary techniques have turned to tools like artificial neural networks, fuzzy logic, and heuristics. In fact, parameters of PID controllers are now increasingly tuned using fuzzy logic and heuristic approaches (Singh and Joshi 2017).

Sinha and Tiwari (2017) presents an effective algorithm for synchronous control of the motion of permanent magnet linear DC motors. This algorithm is based on a PID controller with appropriately adjusted parameters to achieve accurate path tracking and reach specified positions. Cross-coupling control was used to reduce synchronous position error on the axes due to disturbances.

Based on their simple design and control principles, PID controllers are extensively used in manufacturing setups. However, significant parameter changes can potentially destabilize the system. This study delves into intelligent tuning strategies, such as the GA, ant colony optimization, and particle swarm optimization. These strategies are coveted by control engineers in their pursuit of automated tuning processes. By selecting the right tuning constants, these techniques bolster control performance, enhancing steady-state characteristics and performance indexes (Divya and Nirmalkumar 2018).

Guzey *et al.* (2018) presented a position and speed controller for dual PMDC motors used in a gantry-style machine tool. Each motor in the system is paired with two controllers: a position controller (P controller) and a speed controller (PI controller). To ensure accurate synchronization of the motors along the axes, a cross-coupling technique was used, along with a method to calculate the system parameters.

Wang and Suh (2017) created a system that uses a microprocessor to regulate the rotational speed and position of a DC motor. The system determines the optimum speed at each movement and comprises a D/A converter in addition to an optical encoder that supplies the converter with signals, providing feedback for voltage and position.

Gholap *et al.* (2017) applied the optimal settings for a PID controller using Particle Swarm Optimization, the Genetic Algorithm, and the Simulated Annealing methods. Simulations of chemical and electrical systems were employed to formulate the algorithms. The controller was configured with two distinct fitness functions. To assess the effectiveness of the proposed algorithms, a comparative study was conducted on benchmark problems related to DC motors and coupled tank systems.

Zhang and Wang (2018), presents a speed control system for several synchronized DC motors. This system consists of non-linear PID control, complemented by the use of a cross-coupling technique. The system performance was tested in case of disturbances, and a comparison between conventional PID control and non-linear PID control was made to ascertain the latter's effectiveness.

Ali *et al.* (2019) presents a parallel position control system for two DC motors operating simultaneously with a Hall sensor. Achieving parallel position control requires precise control of both speed and position for these motors. The back EMF compensator and the current model speed observer are used to estimate speed and position. Speed can be estimated using the average speed between the speed observer and the actual speed, while the position error between the motors can be compensated using the instantaneous position compensator.

According to (Bharat *et al.* 2019) two predominant schools of thought on PID tuning, the classical and the contemporary, are delved into. Popularity has been garnered by PID control due to its simplicity in understanding, application, and study. Both the traditional and the state-of-the-art methods for optimizing PID settings are examined in this work. Although the primary focus is placed on control system and biomedical applications, a comprehensive understanding of PID control across various contexts is aimed to be provided, extending from classical PID to its integration with intelligent control.

Pinto *et al.* (2020), introduces a new system for synchronous velocity control and precise position tracking using nonlinear time-frequency. The controller is based on wavelet transformation and adaptive control. The performance of this proposed system is tested against PID and fuzzy controllers for comparative purposes.

Bae *et al.* (2020), employs the leader-follower (or master-slave) method to control the synchronized speed of two DC drives. In this method, one motor is the main drive that receives the reference signal and tracks the required trajectory, while the second motor (the slave) follows the main motor's movement. However, this method has an issue: if disturbances affect the slave motor or if it encounters any problem, the main motor does not stop working.

3. MATERIALS AND METHODS

3.1 Permanent Magnet DC Motor

PMDCs have numerous applications and are distinguished from conventional DC motors primarily by the absence of field coils. They operate on the same principle as shunt-connected motors. However, in PMDCs, the field flux is produced by the permanent magnet itself, rather than by field windings. Despite most small DC motors being of the PMDC type, they generally produce less power than the converted type DC motors. Due to their lower flux compared to motors with field windings, PMDCs often run at higher speeds but provide less torque (Namazov and Basturk 2010, Dorf and Bishop 2011).

PMDCs are a subset of DC motors. Figure 3.1 illustrates the two main components of this motor: the stator, which is a steel cylinder with magnetic poles arranged so that they face the armature, and the rotor, also known as the armature (Chau and Wang 2011).

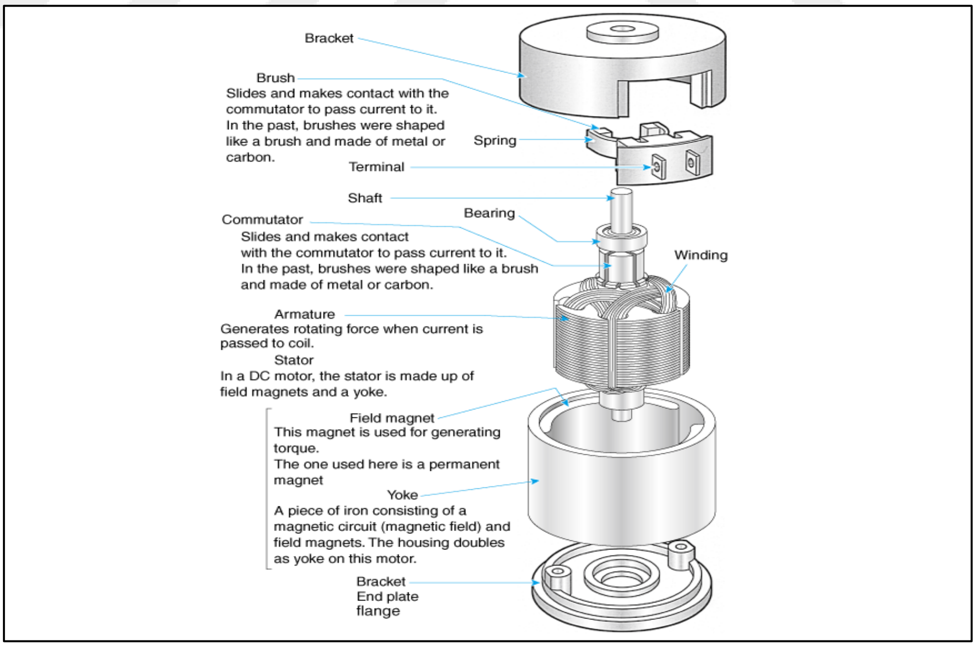

Bracket
Brush
Slides and makes contact with the commutator to pass current to it. In the past, brushes were shaped like a brush and made of metal or carbon.
Spring
Terminal
Shaft
Bearing
Commutator
Slides and makes contact with the commutator to pass current to it. In the past, brushes were shaped like a brush and made of metal or carbon.
Winding
Armature
Generates rotating force when current is passed to coil.
Stator
In a DC motor, the stator is made up of field magnets and a yoke.
Field magnet
This magnet is used for generating torque.
The one used here is a permanent magnet
Yoke
A piece of iron consisting of a magnetic circuit (magnetic field) and field magnets. The housing doubles as yoke on this motor.
Bracket
End plate
flange

Figure 3.1 Construction of PMDC motor (Chau and Wang 2011)

The installation of these poles inside the cylinder and is arranged as shown in Figure 3.2 (a and b) when (a) represents 2-pole PMDC and (b) represents a 4-pole PMDC so that

The north (N) pole is fixed, then the south (S) pole, then the north pole, and so on. As for the rotor it consists of commutator segments, brushes, and winding slots as in conventional machines (Chau and Wang 2011, Mohan 2003).

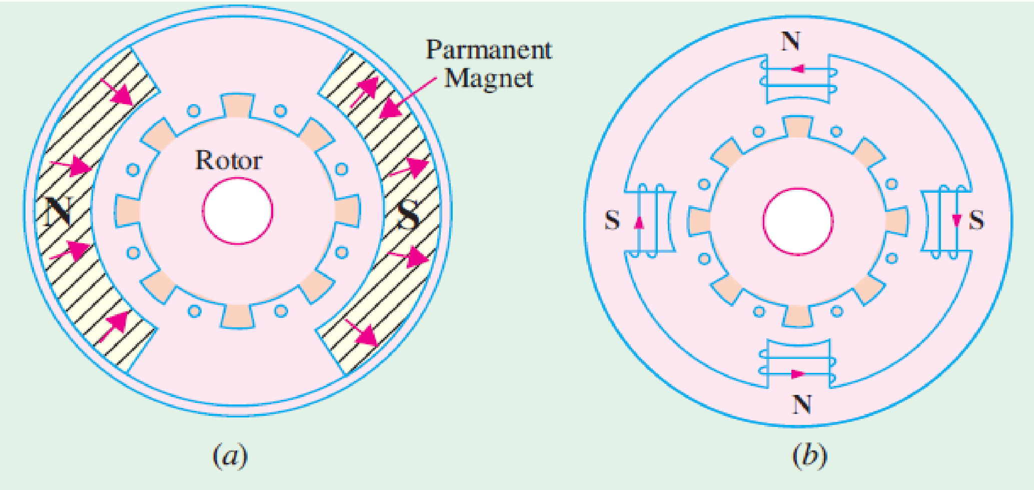


Figure 3.2 Arrangement of connecting the magnetic poles inside the stator (a) represents 2-pole PMDC and (b) represents a 4-pole PMDC (Chau and Wang 2011)

A PMDC operates on the same principle as a conventional DC motor. The direction of the mechanical force exerted on a conductor when introduced to a magnetic field is determined by Fleming's Left Hand Rule. This concept can be expressed by the following Equation (3.1) (El-Sharkawi 2015).

$$F = BIL \tag{3.1}$$

Where F denotes the Mechanical Force, measured in Newton, B represents the strength of the magnetic field, measured in Tesla (weber/m^2), I corresponds to the current flowing through the conductor, measured in Ampere, and L is the length of the conductor, measured in meters.

The interplay of these forces results in a torque that propels the rotation of the armature. These motors are compatible with a range of DC voltages, including 6 volts, 12 volts, and 24 volts. These voltages can either be directly sourced from DC batteries or indirectly from an AC source that is then converted to DC using a rectifier circuit. The torque of these motors is produced when conducting wires interact with the magnetic flux from the permanent magnet (Chau and Wang 2011, Mohan 2003).

The most important of advantages and drawbacks of PMDC can be summarized as (Chau and Wang 2011, Mohan 2003).

- Advantages:

1) Low manufacturing cost: In PMDC motors with very little power, they have a low manufacturing cost due to the use of permanent magnet excitation instead of field coils.
2) Small physical size: The size of PMDC motors is often smaller than the field coil motors, although they have the same power.
3) High efficiency: From the torque-speed performance curve of the PMDC motor shown in Figure 3.3, the efficiency of the PMDC motor is greater than that of the wound-field motors because the flux is not required and therefore there are no copper losses.
4) Low noise generation: PMDC motors operating at low voltages produce little noise in the air.

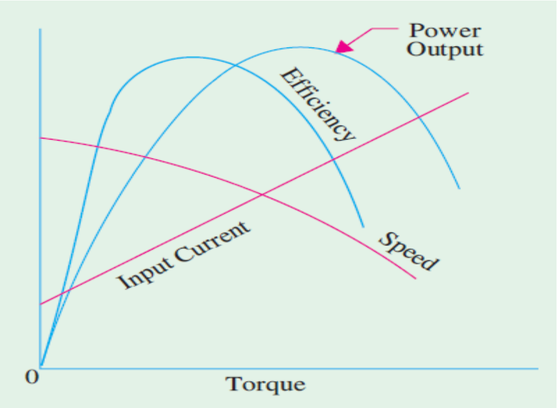


Figure 3.3 PMDC speed– torque curve (Chau and Wang 2011)

- Disadvantages:

1) High temperature: The temperature of these motors rises during work because they are designed completely closed to prevent the magnets of these motors from attracting magnetic waste from the surrounding area, as the magnetic field of these motors is always active even if the motor is not running.
2) Sudden failure: The most serious disadvantage in these motors is to stop working due to the removal of the permanent magnet through the reaction of the Magnetic Motive Force (MMF) in the armature. The removal of permanent magnets occurs due to several reasons, such as:-

a) Rise in motor current results from a fault or transient or contact in the armature circuit.
b) The effect of temperature.
c) Improper brush shift.
d) Improper design.

3.1.1 PMDC mathematical model

PMDC motor uses permanent magnets to generate the magnetic field, rather than relying on external coils. For efficient control and design of PMDC motors, understanding their mathematical model is crucial. The mathematical model provides a clear understanding of how different parameters and variables relate and affect the motor's performance. In Figure 3.4, the analogous circuit for a DC motor with a permanent magnet is illustrated. An armature coil is connected to a voltage source. As the magnetic coil rotates and intersects the flux paths generated by the permanent magnet, a back emf (Ea) is produced, opposing the direction of the voltage source. This back emf, along with the inductance (La) and the armature resistance (Ra), constitutes the electrical component of the circuit. The mechanical component comprises the moment of inertia (Jm) and a viscous coefficient of friction (Bm). The circuit also includes two other parameters: Kt, the torque constant, and Kv, the back emf constant.

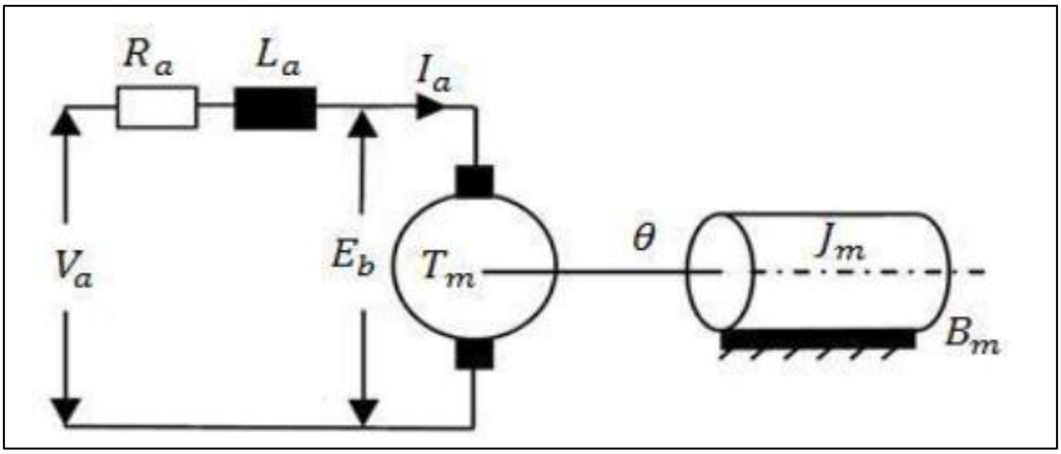


Figure 3.4 Equivalent circuit of PMDC (Adel *et al.* 2018)

The PMDC block diagram is shown in Figure 3.5

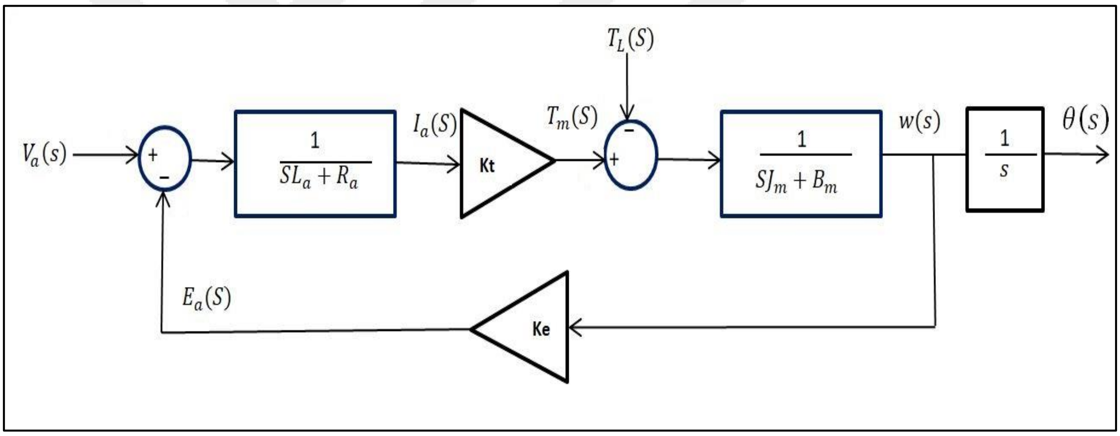


Figure 3.5 Block diagram of PMDC (Adel *et al.* 2018)

The mathematical model for the function in Figure 3.5 can be listed in the following Equation from (3.2) to Equation (3.5) (Adel *et al.* 2018, Rathod *et al.* 2003).

$$v_a(t) = e_a(t) + R_a i_a(t) + L_a \frac{d}{dt} ia(t) \tag{3.2}$$

$$e_a\,(t) = k_e\ \omega_m(t) \tag{3.3}$$

$$T_m(t) - T_L = J_m \frac{d}{dt} \omega_m(t) + B_m \omega_m(t) \tag{3.4}$$

$$T_m(t) = K_t i_a(t) \tag{3.5}$$

By using Laplace transformation for the Equations (3.2) to (3.5), we obtain Equations from (3.6) to Equation (3.9)

$$V_a\,(s) = E_a(s) + \ R_a\, I_a(s) + SL_a I_a(s) \tag{3.6}$$

$$E_a(s) \ = \ k_e \ \omega_m(s) \tag{3.7}$$

$$T_m(s) - T_L = SJ_m\omega_m(s) + B_m\omega_m(s) \tag{3.8}$$

$$T_m(s) = K_t I_a(s) \tag{3.9}$$

Equations (3.10) and (3.11) define the global transfer functions for speed and position control of the PMDC motor.

$$\frac{\omega_m(s)}{V_a(s)} = \frac{k_e}{JL_aS^2+(JR_a+BL_a)S+BR_a+k_e^2} \tag{3.10}$$

$$\frac{\theta(s)}{V_a(s)} = \frac{k_e}{JL_aS^3+(JR_a+BL_a)S^2+\left(BR_a+k_e^2\right)S} \tag{3.11}$$

Where k_e equal to K_t and:

The key parameters of the system under consideration include R_a, the armature resistance measured in Ohms (Ω); V_a, the nominal voltage in volts (V); K_v, the back EMF (electromotive force) constant in volt-seconds per radian (Vsec/rad); L_a, the armature inductance in Henry (H); J_m, the moment of inertia in kilogram-meters squared (kg.m²); B_m, the friction coefficient in Newton-meter-seconds per radian (Nm.s/rad); E_a, the electromotive force or back EMF in volts (V); T_L, the nominal load torque in Newton-meters (Nm); ω_m, the motor speed in radians per second (rad/sec); θ(s), the actual position in radians (rad); and K_t, the torque constant in Newton-meters per Ampere (Nm/A).

3.2 Design Speed – Position Control System

The type of the motor used in this thesis is the PMDC. In this chapter, the main features of this motor is explained such as including its modeling, design, and applications, as well as its advantage and disadvantage. The PMDC's speed and location are the focus of the design and execution of a cascade control system. To get the highest possible performance, we fine-tune the controller improvements using optimization methods. Our optimization algorithms are directed by a benchmark called integral time absolute error, which measures how well they are doing at closing the gap between the real and reference signals.

3.3 Structure of Cascaded PI Controller with PMDC

The structure of the system is shown in Figure 3.6 and it consists of the control circuit, the power circuit, and the PMDC motor circuit, all of which are coupled to one another. A cascade of PID controller consisting of three separate controllers is integrated into the control circuit. In the following paragraphs, cascade PID configuration will be illustrated.

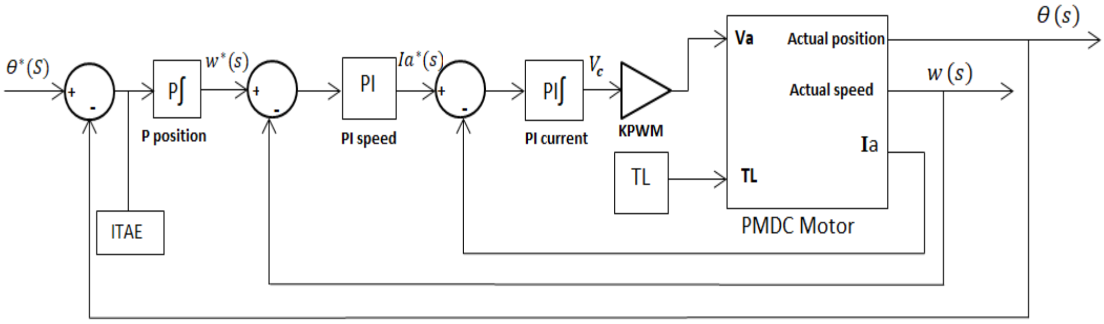


Figure 3.6 Schematic diagram of position-speed control system

The power circuit consists of a 220 volt AC source, and it is converted to a 12 volt DC source using the Rectifier circuit. The output voltage is controlled by a dc-dc converter to suit the motor operation as shown in Figure 3.7 (a, b, c, d and e)

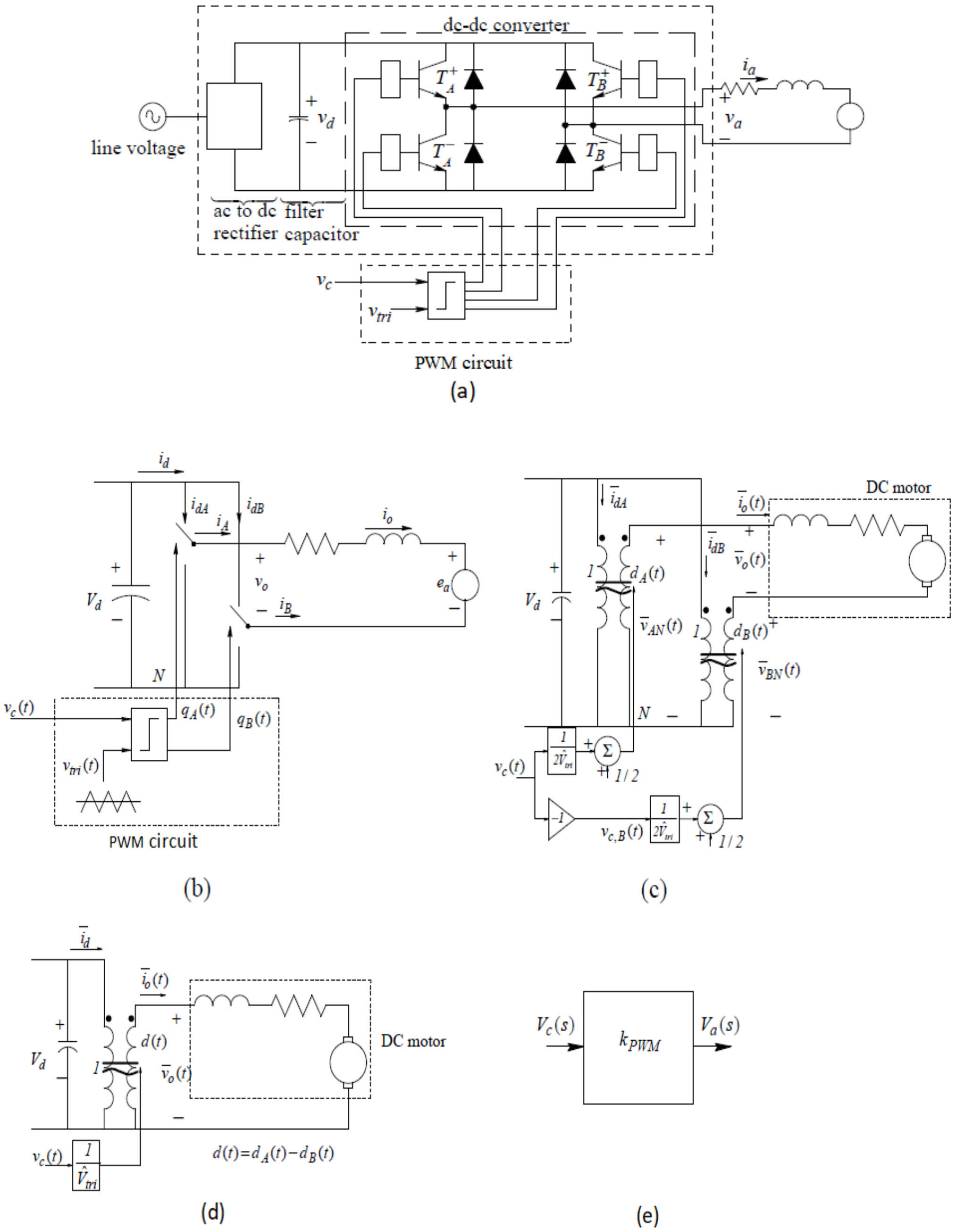


Figure 3.7 (a) Switch-mode converter for dc motor drives; (b), (c) and (d) its simple representation; (e) linearized representation (Parpinelli *et al.* 2011)

The voltage control V_c comes from the PI speed controller. For pole A ($V_c(t) = V_{c,A}(t)$) and for pole B ($V_c(t) = -V_{c,B}(t)$) these control voltages are compared with the same triangular waveform generated inside the PWM circuit resulting in the switching functions $q_A(t)$ and $q_B(t)$, In addition to the duty ratios for the two poles shown in Equation (3.23) and Equation (3.24).

$$d_A(t) = \frac{1}{2} + \frac{1}{2}\frac{v_c(t)}{V_{tri}^{\hat{n}}} \tag{3.23}$$

$$d_B(t) = \frac{1}{2} - \frac{1}{2}\frac{V_c(t)}{V_{tri}} \tag{3.24}$$

The converter average output voltages, as illustrated in Figure 3.11 (c) can be represented in Equation (3.25) and Equation (3.26).

$$V_{AN} = \frac{V_d}{2} + \frac{V_d}{2V_{tri}^{\wedge}} V_c(t) \tag{3.25}$$

$$V_{BN} = \frac{V_d}{2} - \frac{V_d}{2V_{tri}^{\wedge}} V_c(t) \tag{3.26}$$

As shown in Figure 3.11 (d), the output voltage at the output terminals can be represented in Equation (3.27)

$$V_o(t) = V_{AN}(t) - V_{BN}(t) = \frac{V_d}{V_{tri}^{\wedge}} V_c(t); \frac{V_d}{V_{tri}^{\wedge}} = KPWM \tag{3.27}$$

So that Equation (3.28) be:

$$V_o(t) = KPWM\, V_c(t) \tag{3.28}$$

The gain constant of the PWM converter is denoted as *KPWM*. The Equation (3.29) denoted as inside the Laplace domain illustrates the combination of PWM control and the dc-dc switch-mode converter after the application of linearization. This is visually shown in Figure 3.2(d).

$$KPWM = \frac{V_a(s)}{V_c(s)} \tag{3.29}$$

Where $V_a(s)$ = The input voltage applied to the PMDC motor and it is the same as the output voltage $V_o(s)$.

3.4 Proposed Cascade Control System

In control engineering, the usage of a cascade PID controller system to boost a machine's efficiency is common practice. Cascade controllers are used to dampen or reject shocks and bring systems back to balance. The cascade controller presented in this thesis is made up of three separate controllers. The first controller is in charge of positioning, the second of regulating velocity, and the third of regulating current flow. Position and velocity are inner controllers in the current controller, as stated by (Yazgan *et al.* 2019), who describe the construction of the controller as an outer loop architecture. Alternatives to a PID controller include a P controller for position control, a PI controller for speed control, and a PI controller for current control. Cascade control systems are widely employed in motor drives due to their adaptability. A higher rate of response is required in the inner loop, with the torque loop being the fastest and the position loop being the slowest. In addition to these advantages, the PMDC motor and power electronic converter are protected by the cascade control system because of its capacity to impose limits on the reference signals used by the system. The speed reference is limited by this theory so that it does not go over the motor's specified maximum speed. In addition, the PWM controller places constraints on the PI current by limiting the reference voltage to 5 volts as it leaves the PI circuit. When compared to a triangle voltage, the aforementioned restriction represents the control voltage (vc).

3.4.1 Current controller

The inner loop as shown in Figure 3.8 consists of a PI controller where the input reference current enters it from the output of the PI speed controller.

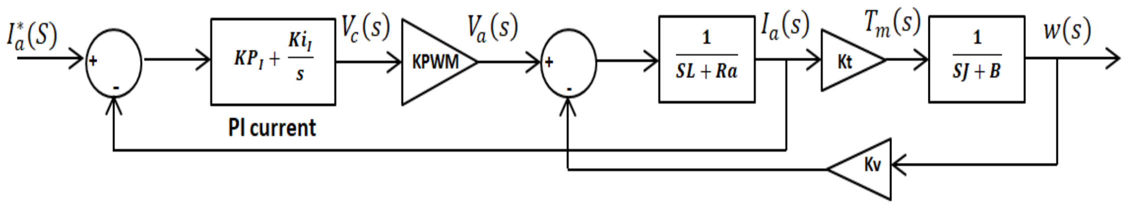


Figure 3.8 Inner (current) loop

The inner loop in Figure 3.9 and Figure 3.10 is simplified by ignoring the load torque.

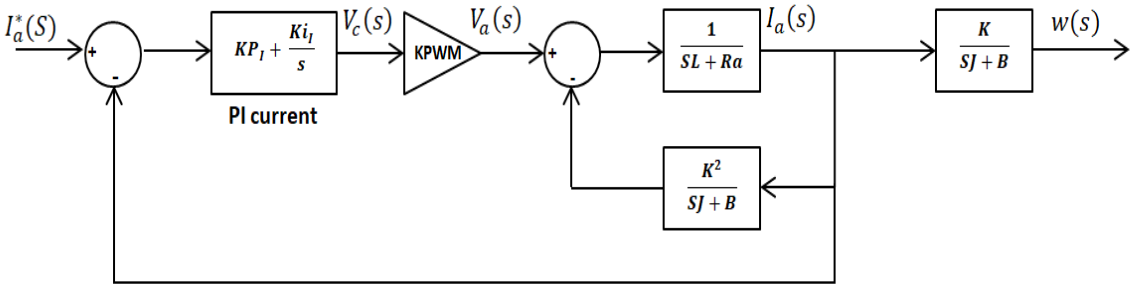


Figure 3.9 First-stage loop simplification

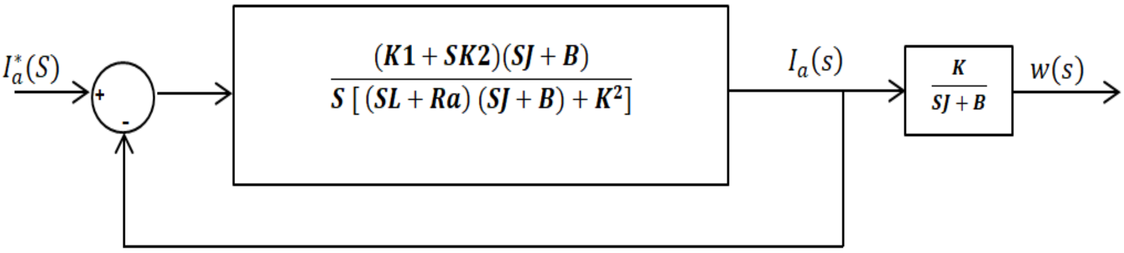


Figure 3.10 Second-stage loop simplification

So, the calculation of K1 shown in Equation (3.30), Equation (3.31) and Equation (3.32)

$$K = K_t = K_v \tag{3.30}$$

$$K1 = KPWM * Ki_I \tag{3.31}$$

$$K2 = KPWM * Kp_I \tag{3.32}$$

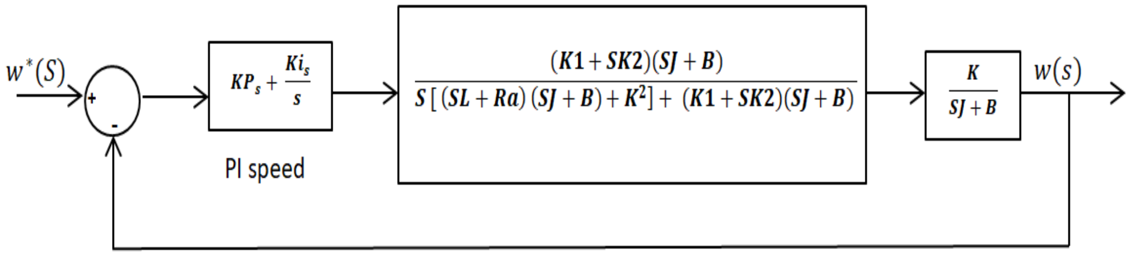
Equation (3.33), which represents the inner loop transfer function, is as follows.

$$\frac{I_a(s)}{I_a^*(s)} = \frac{(K1+SK2)(SJ+B)}{S\,[\,(SL+Ra)\,(SJ+B)+K^2]+(K1+SK2)(SJ+B)} \tag{3.33}$$

3.4.2 Speed controller

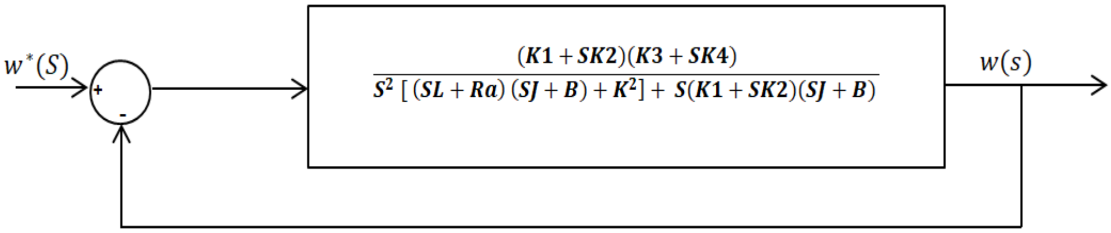
Figure 4.6 demonstrates that the speed reference signal is transmitted to the speed loop via the output PI controller of the P position controller.

The can be simplified in Figure 3.11 and Figure 3.12



$w^*(S)$
+
-
$KP_s + \frac{Ki_s}{s}$
PI speed
$\frac{(K1+SK2)(SJ+B)}{S\,[(SL+Ra)\,(SJ+B)+K^2]+\,(K1+SK2)(SJ+B)}$
$\frac{K}{SJ+B}$
$w(s)$

Figure 3.11 First-stage loop simplification



$w^*(S)$
+
-
$\frac{(K1+SK2)(K3+SK4)}{S^2\,[\,(SL+Ra)\,(SJ+B)+K^2]+\,S(K1+SK2)(SJ+B)}$
$w(s)$

Figure 3.12 Second-stage loop simplification

So, to find *K3* and *K4* form Equation (3.34) and Equation (3.35)

$$K3 = K * \; Ki_s \tag{3.34}$$

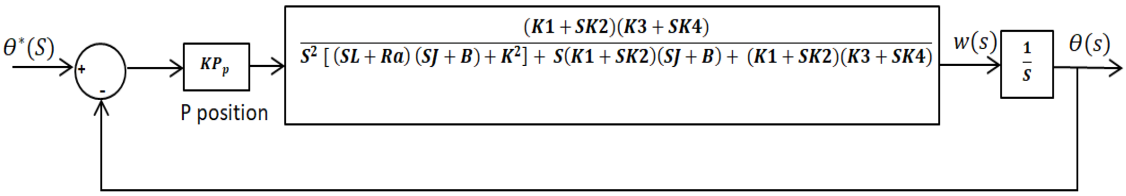
$$K4 = K * \; KP_s \tag{3.35}$$

The speed loop transfer function can be represented by the Equation (3.36).

$$\frac{w_{(s)}}{w^*_{(s)}} = \frac{(K1+SK2)(K3+SK4)}{S^2[(SL+Ra)(SJ+B)+K^2]+S(K1+SK2)(SJ+B)+(K1+SK2)(K3+SK4)} \qquad (3.36)$$

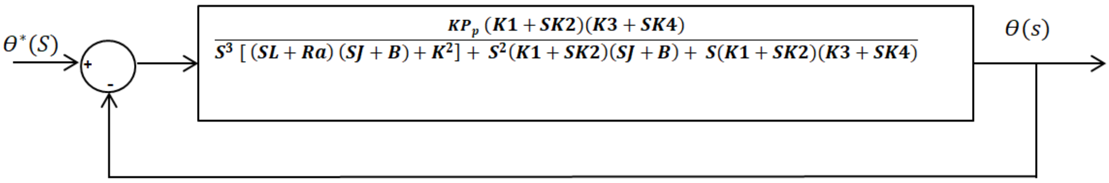
3.4.3 Position controller

The position loop can be simplified in Figure 3.13 and Figure 3.14



$\theta^*(S)$
+
-
KP_p
P position
$\frac{(K1+SK2)(K3+SK4)}{S^2\,[(SL+Ra)\,(SJ+B)+K^2]+\,S(K1+SK2)(SJ+B)+\,(K1+SK2)(K3+SK4)}$
$w(s)$
$\frac{1}{s}$
$\theta(s)$

Figure 3.13 Simplification position loop step one



$\theta^*(S)$
+
-
$\frac{KP_p\,(K1+SK2)(K3+SK4)}{S^3\,[(SL+Ra)\,(SJ+B)+K^2]+\,S^2(K1+SK2)(SJ+B)+\,S(K1+SK2)(K3+SK4)}$
$\theta(s)$

Figure 3.14 Simplification position loop step two

Equation (3.37) represents the position looping transfer function.

$$\frac{\theta_{(S)}}{\theta^*_{(S)}} = \frac{KP_p(K1+SK2)(K3+SK4)}{S^3[(SL+Ra)(SJ+B)+K^2]+S^2(K1+SK2)(SJ+B)+S(K1+SK2)(K3+SK4)+KP_p(K1+SK2)(K3+SK4)} \qquad (3.37)$$

3.5 Performance Metrics

Performance requirements for today's complicated control systems are often more nuanced than those discussed thus far. The timing of an error is a crucial component that must often be taken into account with the error itself. An objective function is a lone indicator of system effectiveness that highlights crucial features of the response. In optimal control theory and linear-state-variable-feedback estimator design, the system is developed to maximize this performance index within a set of restrictions; here is where the concept of an objective function comes into play. The numerous error and/or time functions serve as performance indices. These aims may be expressed as a variety of integrals, including integrals of absolute error (IAE), integrals of squared error (ISE), integrals of time (ITAE), and integrals of time (ITSE), among others.

In this thesis the integral time absolute error (ITAE) is used. The ITAE criterion generally produces a smaller overshoot and oscillation than ISE and IAE criteria. In addition, it is the most sensitive of the three, and sometimes too sensitive-slight parameter variation degrades system performance. ITAE can be represented by the following Equation (3.54):

$$\text{ITAE} = \int_0^{T_s} t|e_\theta(t)|dt + \int_0^{T_s} t|e_w(t)|dt + \int_0^{T_s} t|e_{Ia}(t)|dt \qquad (3.54)$$

Within the context of this study, Ts signifies Simulink time, $e_\theta$indicates the position error, which is the difference between the reference and actual position, e_w is the speed error, computed as the difference between the reference and actual speed, and e_{Ia} is the current error, computed as the difference between the reference and actual current. The proportional gain K_p and integral gain K_i are subject to the constraints Kp = $KP_{LB} \leq KP \leq KP_{UB}$ and Ki = $Ki_{LB} \leq Ki \leq Ki_{UB}$, respectively. LB and UB represent the lower and upper bounds for the parameters used in the optimization code. It's noteworthy that a particular fitness equation is employed in this study to minimize overshoots. Also, the fitness functine = ITAF+ steady state error+ overshoot.

Figure 3.15 shows the effect of the objective function and its use with the BOA algorithm, where after updating the PI controller parameters, the error is compared between the reference values and the actual values at each iteration until to get best results.

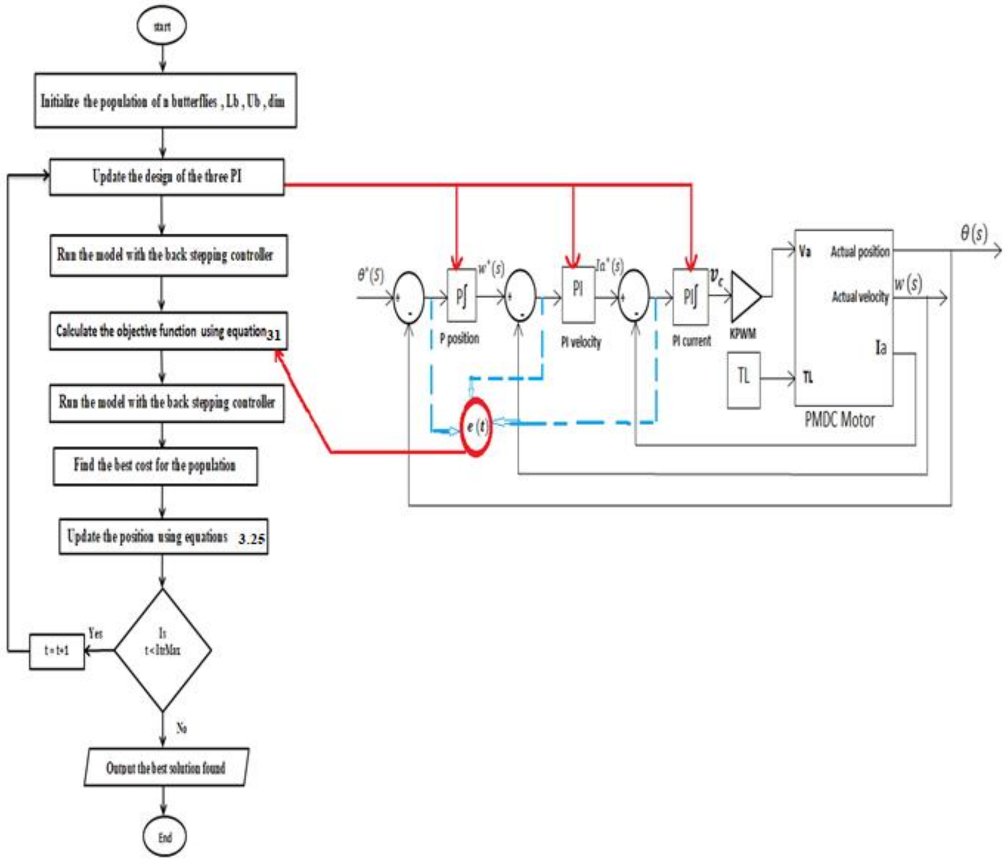

start
Initialize the population of n butterflies , Lb , Ub , dim
Update the design of the three PI
Run the model with the back stepping controller
Calculate the objective function using equation 31
Run the model with the back stepping controller
Find the best cost for the population
Update the position using equations 3.25
Is t<ItrMax
Yes
t=t+1
No
Output the best solution found
End
θ*(S)
P∫
P position
w*(s)
PI
PI velocity
Ia*(s)
PI∫
PI current
v_c
KPWM
TL
Va
TL
Actual position
Actual velocity
Ia
PMDC Motor
θ(s)
w(s)
e(t)

Figure 3.15 ITAE effect on the system

3.6 Optimization Tuning Methods

In order to obtain accurate results, the PID parameters of the system must be carefully selected various techniques can be employed to tune the parameters of PID and implement it in a permanent magnet synchronous (PMSC) motor. These techniques include the Ziegler-Nichols Method (Z-N), the Cohen-Coon Method, artificial neural networks, and fuzzy logic. Optimization methods such as the Particle Swarm Optimization (PSO) (as discussed by (Hasan *et al*. 2014, Arora and Singh 2019), the Butterfly Optimisation Algorithm (BOA), and the genetic algorithm are also applicable (Latif *et al.* 2020, Levine 2019, Kumar *et al.* 2010) etc.

In this study, the classic method (CM) is used and three optimal methods for calculating and extracting the values of the PID cascade parameters which are the GA, DOA, and the

BOA with a comparison between these methods to choose the best results and they will be explained in detail in the following section:

3.6.1 Classical method

In this section, the values of the P and PI parameters of the cascade control system are calculated using Ziegler Nichols method block diagram reduction, and simplifying each loop separately and calculating its parameters. This method assumes some assumptions such as neglecting the load and canceling the back EMF effect (Biswas 2013). Initially, the parameters of the inner loop or the current loop are calculated. Figure 3.16 represents the first control loop. In a PMDC motor, both torque and current are proportional so that the current can be considered as a control.

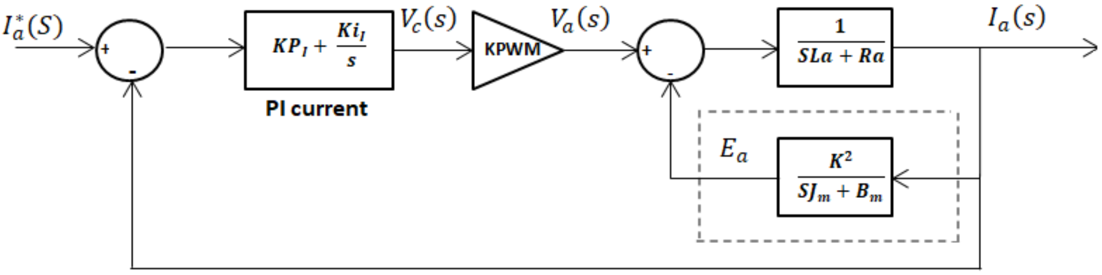
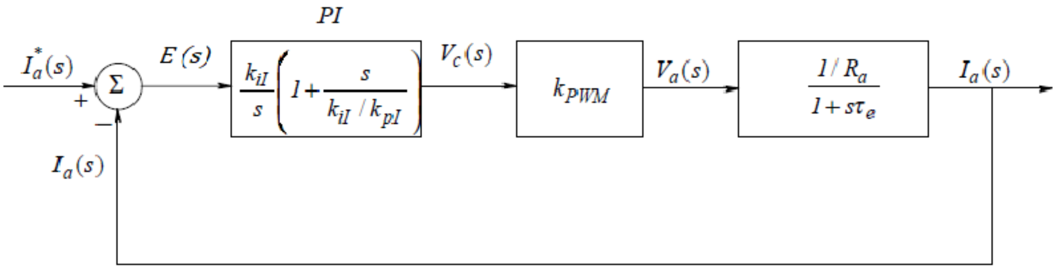


Figure 3.16 Current loop

Owing to the simplifying assumptions adopted in this method (Parpinelli and Lopes 2011, Biswas 2013), the effects of torque load are overlooked, as is the impact of Ea, given the large value of Jm. As a result, Figure 3.17 presents a streamlined current control loop.



PI
$I_a^*(s)$
+
—
Σ
$E(s)$
$\frac{k_{iI}}{s}\left(1+\frac{s}{k_{iI}/k_{pI}}\right)$
$V_c(s)$
k_{PWM}
$V_a(s)$
$\frac{1/R_a}{1+s\tau_e}$
$I_a(s)$
$I_a(s)$

Figure 3.17 Simplified current control loop

The transfer function of the PI current controller as in Equation (3.12)

$$\frac{V_c(s)}{E(s)} = Kp + \frac{K_i}{s} = \frac{K_i}{s}\left(1 + \frac{s}{{Ki}/_{Kp}}\right) \tag{3.12}$$

Figure 3.7 displays a simplified version of the current loop, and its open-loop transfer function, represented as $G_{I\,ol}$, is provided in Equation (3.13).

$$G_{I\,ol} = \frac{k_{iI}}{S}\left(1 + \frac{S}{{k_{iI}}/_{k_{pI}}}\right) * KPWM * \left(\frac{1/_{R_a}}{1+s\tau_e}\right) \tag{3.13}$$

When equating the armature's inductance *La* to the armature resistance *Ra*, we obtain the electrical time constant *e*. In this context, ***kpI*** denotes the proportional increase of the current controller, while ***kiI*** represents the integral gain of the current controller.

The motor pole, expressed in an equation, is cancelled out using Equation (3.14) as referenced in Equation (3.12).

$$\frac{k_{iI}}{k_{pI}} = \frac{1}{\tau_e} \qquad \text{or} \qquad k_{pI} = \tau_e * k_{iI} \tag{3.14}$$

As seen in Equation (3.15), the elimination of the pole in the motor transfer function results in the following:

$$G_{I\,ol} = \frac{k_{iI}}{S}\left(1 + \frac{S}{{k_{iI}}/_{k_{pI}}}\right) * KPWM * \left(\frac{1/_{R_a}}{1+s\tau_e}\right) \; OR \; G_{I\,ol} = \frac{k_{iI}}{S}\left(1 + \frac{S}{1/_{\tau_e}}\right) * KPWM * \left(\frac{1/_{R_a}}{1+s\tau_e}\right) \tag{3.15}$$

Finally, the range of the crossover frequency *ωci* for the current loop can be described using the open-loop transfer function, as presented in Equation (3.16).

$$\omega\text{ci} = \frac{KPWM*k_{iI}}{R_a} \text{ or } k_{iI} = \frac{\omega\text{ci}*R_a}{KPWM} \tag{3.16}$$

or it can be calculated from the following Equation (3.17)

$$\omega ci = 2\pi f ci \tag{3.17}$$

and its value can be selected about ten times smaller than the switching frequency (fci) for DC-DC converter (Parpinelli and Lopes 2011). In this study, the supposed switching frequency is 2 kHz. Equation (3.16) and Equation (3.17) can be utilized to determine the appropriate values for the parameters of the PI current controller. Figure 3.18, based on Biswas (2013), depicts the closed current loop. This loop is considered optimal for design purposes and is represented by unity to ascertain the speed loop parameters..

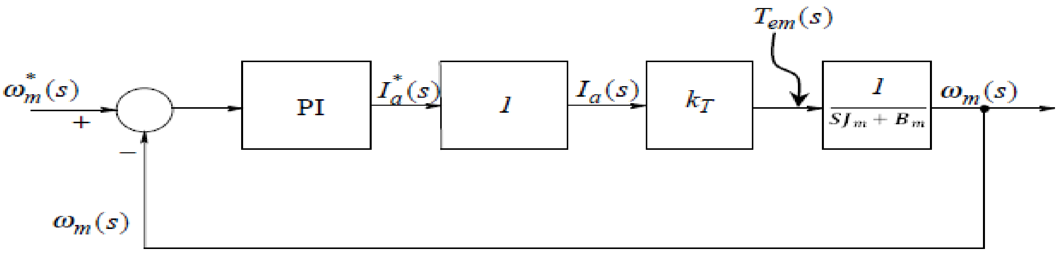


Figure 3.18 Block diagram of the speed loop

The transfer function of this loop, $G_{s\,ol}(s)$) , is given by Equation (3.18).

$$G_{s\,ol}(s) = \frac{k_{iS}}{S}\left(1 + \frac{S}{k_{iS}/k_{pS}}\right) * \left(\frac{1/B_m}{1+s\tau_m}\right) \tag{3.18}$$

The mechanical time constant, m, is given by the ratio *Jm/Bm*. Equation (3.19) details the motor's inner current control loop and illustrates how the pole in the motor's mechanical section can be cancelled.

$$\frac{k_{iS}}{k_{pS}} = \frac{1}{\tau_m} \tag{3.19}$$

where, K_{is} is represent to the integral gain and, K_{psv} also, represent to the proportional gain of the controller.

Equation (3.20) illustrates the result of setting the bandwidth frequency ωcs of the speed control loop to be a factor of 10 lower than ωci.

$$\omega cs = \frac{k_{is} \cdot k_t}{B_m} \tag{3.20}$$

As can be seen in Figure 3.19, the outermost loop is used for position control.

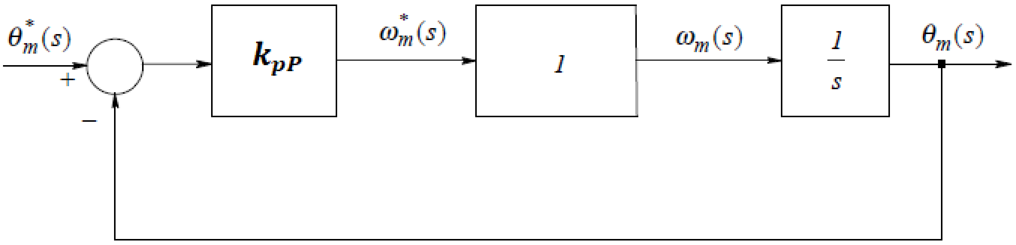


Figure 3.19 Block diagram of position loop (Biswas 2013)

Figure 3.9 from (Biswas 2013) assumes a perfect speed loop to determine the k_{pP} value for the position loop. The open-loop transfer function for position management is provided in Equation (3.21). The k_{pP} value for position, where the bandwidth frequency ωps is set to be a tenth of ωcs, is detailed in Equation (3.22).

$$G_{Pol}(s) = \frac{k_{pP}}{s} \tag{3.21}$$

$$\omega ps = k_{pP} \tag{3.22}$$

3.6.2 Genetic algorithm optimization

Holland initially introduced the concept of a genetic algorithm (GA) in 1970. This strategy has its roots in evolutionary biology and the concept of natural selection. This technique acts as an optimizer initially by picking a fixed number of chromosomes from an initial pool. The fitness function is used to determine how effective each chromosome is in solving a problem. The next generation is the product of natural selection acting on the present population and giving rise to the next set of parents. The current generation is anticipated to find a better answer than its predecessors. This process is repeated over and over again until one of the generations finds the optimal configuration for the system. Figure 3.20 depicts a genetic algorithm flowchart. This is because there are several steps involved in the process.

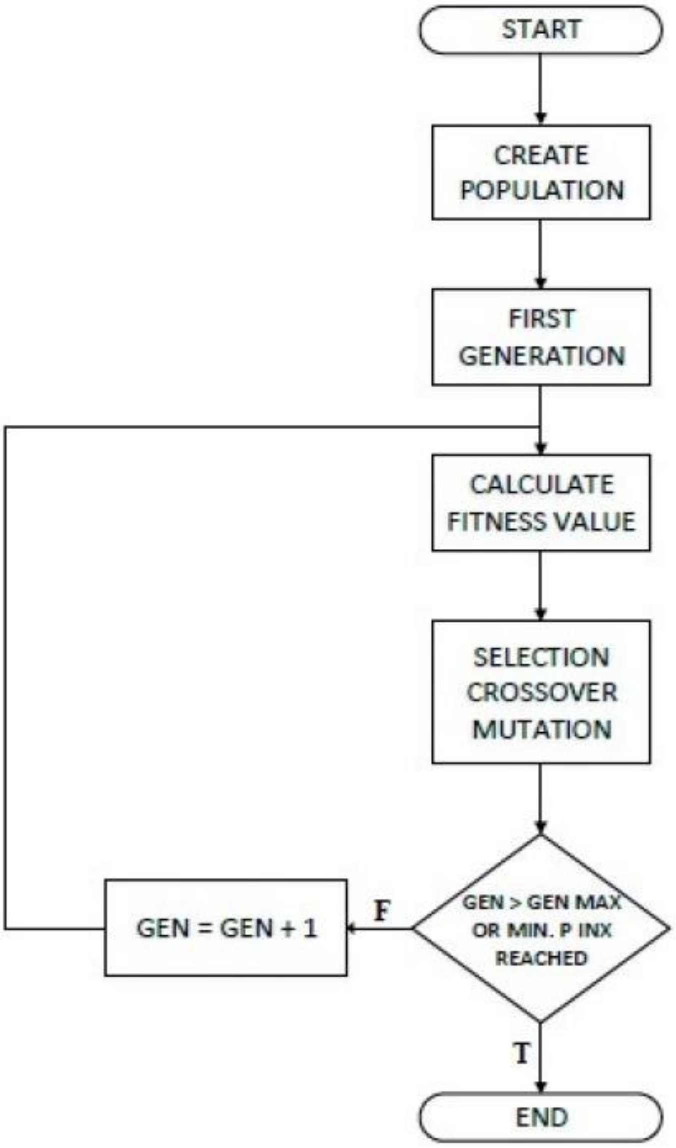


Figure 3.20 Genetic algorithm process flowchart

The GA settings utilized in this analysis are shown in Table 3.1.

Table 3.1 GA Parameters

GA Parameters	**Value**
Generation	100
Population Size	20
No. dimension	5
Crossover	0.6
LU, UB	0.300

3.6.3 Butterfly optimization algorithm

The Butterfly optimization (BOA) is a novel metaheuristic algorithm described in this study that is inspired by natural systems. This algorithm takes its cues for how to best accomplish its biological goals from the behavior of butterflies as they forage for food and mate. Butterflies may follow the scent of nectar back to the flower that's providing it.

The chemical receptors of a butterfly are the nerve cells that are found on its outer skin. The butterfly's antennae and legs, among other places, are covered with specialized cells called chemical sensors. In other words, the butterfly's scent travels with it as it flits from one location to another, allowing other butterflies to pick up on its presence as it does so (Latif *et al.* 2020). The butterfly's fitness is reflected in the strength of its odor.

The foundational elements of sensation and olfactory processing can be categorized into three main areas:

- The sensory type, denoted as c, has a possible value range of 0 to 1.
- The stimulus intensity is symbolized by I.
- The degree or strength of the stimulus upon which the butterfly relies is termed the power exponent and is symbolized by "a". Its values can span from 0 to 1.

The behavior of butterflies is based on two primary factors: the composition of the scent function and the fluctuation in the aroma's intensity. Both factors influence the natural behavior of butterflies. The fragrance function can be depicted mathematically by considering the product's physical density, as described in Equation (3.38).

$$\mathrm{f} = \mathrm{cI}^{\mathrm{a}} \tag{3.38}$$

where f represents the fragrance's perceived magnitude, or how strongly it is perceived by other butterflies. The variables c, I, and a have been defined above.

If a butterfly can pick up on the pheromones being released by another butterfly, it will travel in that direction. The global search phase may be represented mathematically by Equation (38). When a butterfly is unable to smell the pheromones being released by other butterflies, it will fly in a random pattern. The Equation (3.39), which represents this phase of the search, is called the local search.

$$X_i{}^{t+1} = X_i{}^t + \left(r^2 * g^* - X_i{}^t \right) * f_i \tag{3.39}$$

$$X_i{}^{t+1} = X_i{}^t + \left(r^2 * X_j{}^t - X_k{}^t \right) * f_i \tag{3.40}$$

Where $X_i{}^{t+1}$ denotes the new solution vector, $X_i{}^t$ signifies the current solution vector, and r is a random number in the range of 0 to 1. g refers to the current best solution, t is the iteration number, and i symbolizes a butterfly $X_j{}^t$& $X_k{}^t$ are the jth and kth butterflies from the solution space, and $\mathrm{f_i}$ signifies the perceived magnitude of the fragrance.

Depending on their requirements, the butterflies may either search close to home or go far out in search of food or a spouse. The butterfly may choose between two different types of searching, allowing it to either go in the direction of the butterfly with the best global search or to travel at random. We call the mechanism by which one may toggle between local and global search a switch probability ρ. The BOA settings for this analysis are shown in Table 3.2.

Table 3.2 The BOA algorithm parameters

Parameters	**Value**
Max Iteration	5.0
No.Search agents	20.0
No. Dimension	5.0
Switch Probability $\boldsymbol{\rho}$	0.80
Power exponent a	0.10
Sensory modality	0.010
LB , UB	0.3000

The butterfly optimization technique is able to be depicted in the form of an approximate flowchart, which may be found in Figure 4.21.

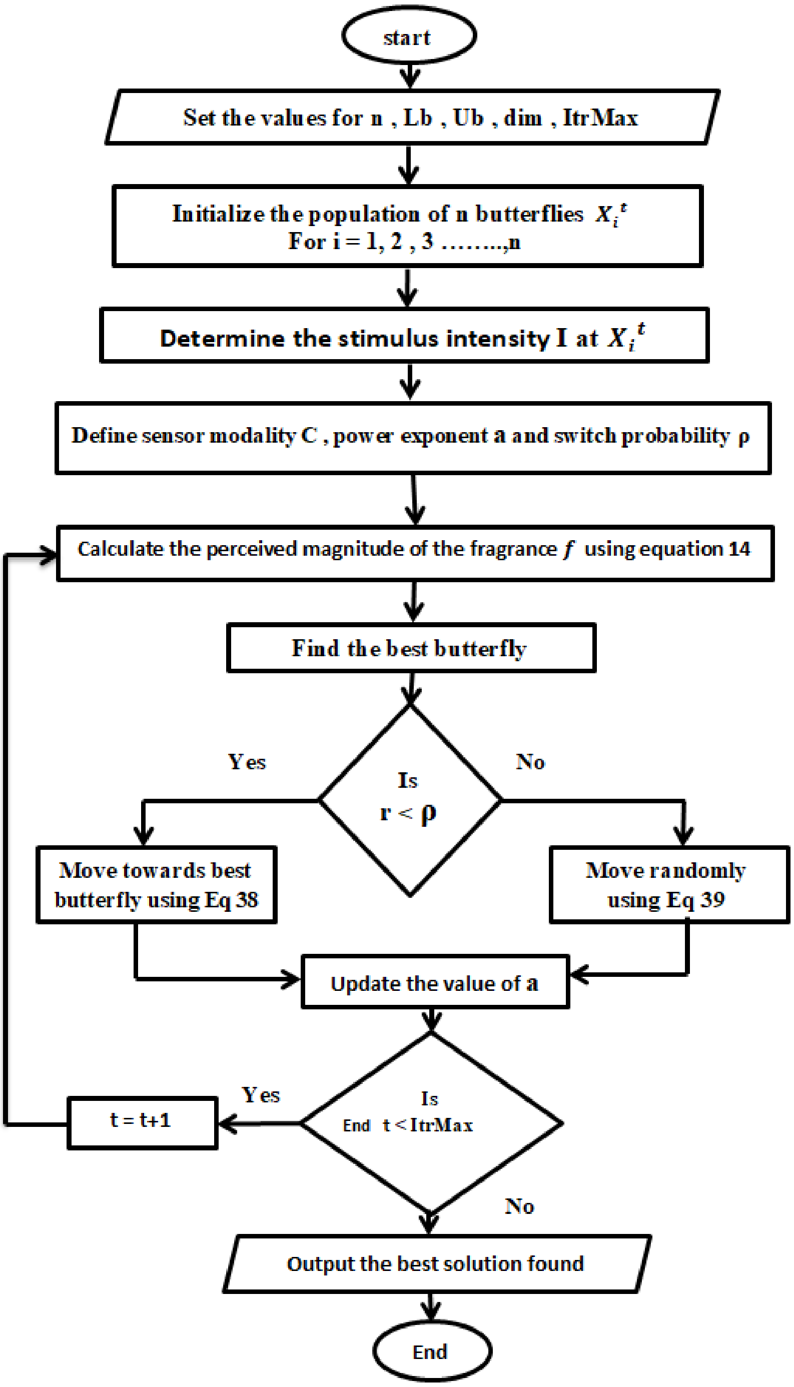

start
Set the values for n , Lb , Ub , dim , ItrMax
Initialize the population of n butterflies X_i^t
For i = 1, 2 , 3 ……..,n
Determine the stimulus intensity I at X_i^t
Define sensor modality C , power exponent a and switch probability ρ
Calculate the perceived magnitude of the fragrance f using equation 14
Find the best butterfly
Yes
No
Is
r < ρ
Move towards best butterfly using Eq 38
Move randomly using Eq 39
Update the value of a
Yes
t = t+1
Is
End t < ItrMax
No
Output the best solution found
End

Figure 3.21 BOA Flowchart

3.6.4 Dandelion optimizer algorithm

The mathematical formulae used in Dandelion Optimization Algorithm DOA are fully explained in this section. In the first section, we learn the mathematical formulae for the two basic weather states. After that, it dives into the math behind the descent and landing phases.

1) Initialization

DOA, like other metaheuristic algorithms inspired by nature, undergoes population evolution and iterative optimization based on population initialization. In the DOA algorithm, each dandelion seed is assumed to represent a potential solution, and the population is expressed in Equation (3.41).

$$population = \begin{bmatrix} x_1^1 & \cdots & x_1^{Dim} \\ \vdots & \ddots & \vdots \\ x_{pop}^1 & \cdots & x_{pop}^{Dim} \end{bmatrix} \tag{3.41}$$

The population size is denoted by "pop" and the dimension of the variable is represented by "Dim". Each potential answer is somewhere between the problem's upper limit *UB* and lower bound *LB*. Expression X_i of the ith person is generated randomly according to the Equation (3.42).

$$X_i = rand \times (UB - LB) + LB \tag{3.42}$$

In the Equation (4.20), i is an integer that ranges from 1 to "pop", and "rand" represents a randomly generated number between 0 and 1. The lower bound (LB) and upper bound (UB) values are expressed as in Equation (3.43) and Equation (3.44).

$$LB = [lb_1, \dots \dots \dots \dots \dots ., lb_{Dim} \tag{3.43}$$

$$UB = [ub_1, \dots \dots \dots \dots \dots ., ub_{Dim} \tag{3.44}$$

During the startup process, DOA chooses its inaugural elite based on the person with the highest fitness score. In this person, the dandelion seed has the best chance of flourishing.

For instance, if the smallest possible number is used, the initial elite Xelite may be expressed mathematically as in Equation (3.45) and Equation (3.46).

$$f_{best} = min(X_i) \tag{3.45}$$

$$X_{elite} = X\left(find\left(f_{best} == (X_i)\right)\right) \tag{3.46}$$

The expression fin() refers to two indices that have the same value.

2) Rising stage

To disseminate from their parent plant, dandelion seeds must first complete the ascending stage. A variety of environmental variables, including wind velocity and humidity, influence the dispersion of the seeds. The following section details the differences between two weather situations.

In the first scenario, the winds are assumed to blow at a lognormal rate on a calm day. This distribution is more likely to produce random numbers spread out along the Y-axis, increasing the chances that dandelion seeds will disperse to distant locations. As a result, the DOA prioritizes exploration. Dandelion seeds drift across the search area, settling wherever the wind carries them. The altitude a dandelion seed reaches depends on the wind's speed. Stronger winds blow the seeds further and to greater heights. Due to the vortices above them continually adjusting to the wind's velocity, dandelion seeds ascend in a spiral pattern. This situation is described by Equation (3.47), the relevant mathematical.

$$X_{i+1} = X_t + \alpha * v_x * v_y * lnY * (X_s - X_t) \tag{3.47}$$

In iteration t, the dandelion seed's location is denoted by X_t, whereas the randomly chosen search site is denoted by X_s. The formula for determining the randomly generated location is given by Equation (3.48).

$$X_s = rand(1, Dim) \times (UB - LB) + LB \tag{3.48}$$

The term "ln Y" refers to a lognormal distribution where the mean is 0 and the variance is 1. The Equation (3.49) used to express this distribution mathematically is:

$$\ln Y = \begin{cases} \frac{1}{y\sqrt{2\pi}} exp\left[-\frac{1}{2\sigma^2}(\ln y)^2\right] & y \geq 0 \\ 0 & y < 0 \end{cases} \tag{3.49}$$

Equation (4.28) uses the symbol y to represent a standard normal distribution with a mean of 0 and a variance of 1. The parameter α is a value that can be adjusted to change the length of the search step, and its formula is included in the Equation (3.50).

$$\alpha = rand(\,) * \left(\frac{1}{T^2}t^2 - \frac{2}{T}t + 1\right) \tag{3.50}$$

Figure 3.22 illustrates the changing value of α over time as the number of iterations increases. The graph shows that α is a variable that randomly fluctuates between 0 and 1, and then gradually decreases towards 0. These fluctuations are useful because they encourage the algorithm to focus on exploring the global search space early on and then switch to a more local search later, which helps ensure accurate convergence after a full global search. vx and vy are coefficients that describe the lift generated by a dandelion as it moves through the air. Equations (3.51) to Equation (3.53) is used to calculate the force acting on the variable dimension.

$$r = \frac{1}{e^{\theta}} \tag{3.51}$$

$$v_x = r * \cos\theta \tag{3.52}$$

$$v_y = r * \sin\theta \tag{3.53}$$

The variable θ is a value that is selected randomly from the range between -π and π.

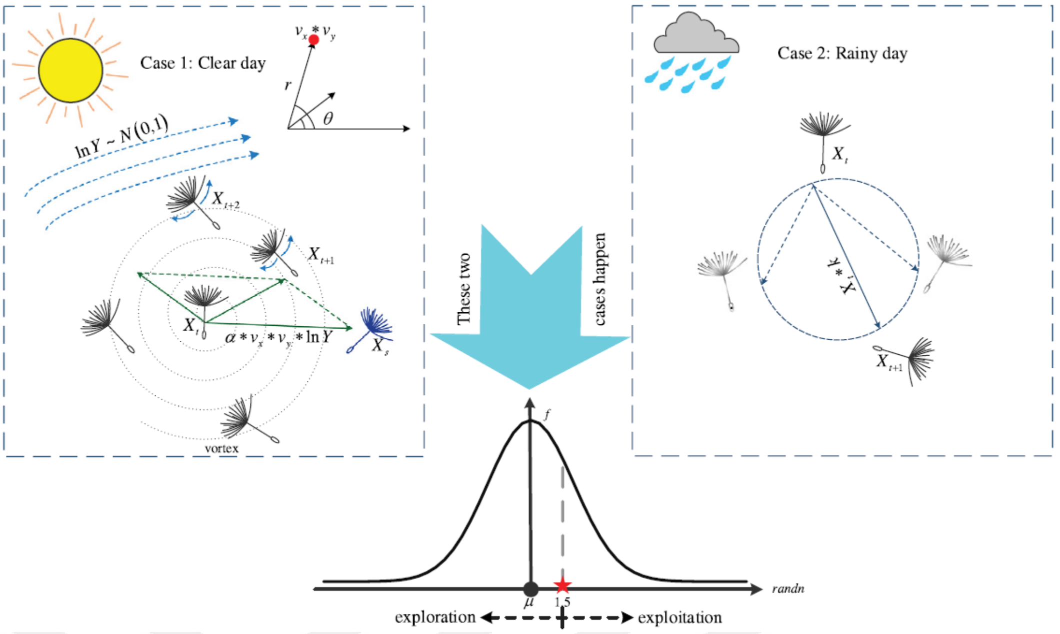


Figure 3.22 Visual representation of rising stage

4. RESULTS AND DISCUSSIONS

This chapter is segmented into two primary sections. The initial section presents both simulation results and real-time outcomes of the cascading P-PI control system, which is utilized to regulate the speed and position of the PMDC motor. The results are based on the values of the parameters obtained through different tuning methodologies, namely CM, GA, BOA, and DOA. The latter section comprises two comparative analyses. The first analysis juxtaposes the four tuning methodologies employed in this thesis: CM, GA, DOA, and BOA. The goal is to find the sweet spot of cascading P-PI controller settings that will allow for the most precise speed and position control for a single-axis PMDC motor.

4.1 Cascade P-PI Results for the PMDC Motor

The PMDC motor's current, position, and speed may all be regulated by the cascade P-PI controller's three separate controllers. A multiple reference position has been implemented to guarantee that the motor completes one or more cycles in order to test the controller's resilience. In addition to testing the controller under no-load and full-load conditions.

4.1.1 Position and speed results based on classical method

The classical method is based on simplifying the block diagram and assumes some assumptions such as canceling the load effect and back emf. In real-time, the effect of the load and the back emf could not be canceled, so this method did not give satisfactory results for controlling the speed and position of the PMDC motor as shown in Figure 4.1, Figure 4.2 and Figure 4.3.

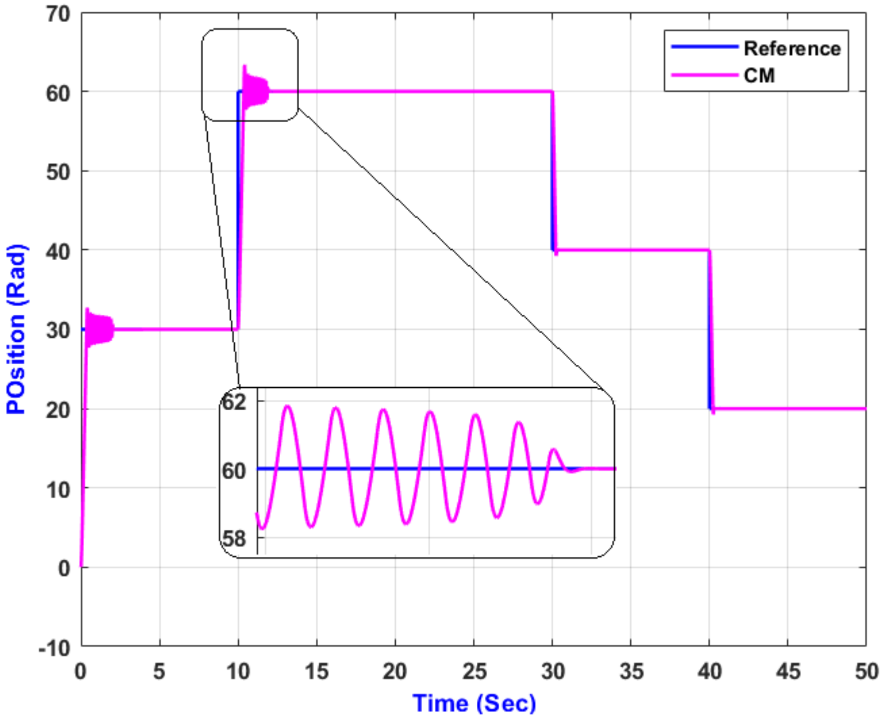


Figure 4.1 Position result based on CM method at no load

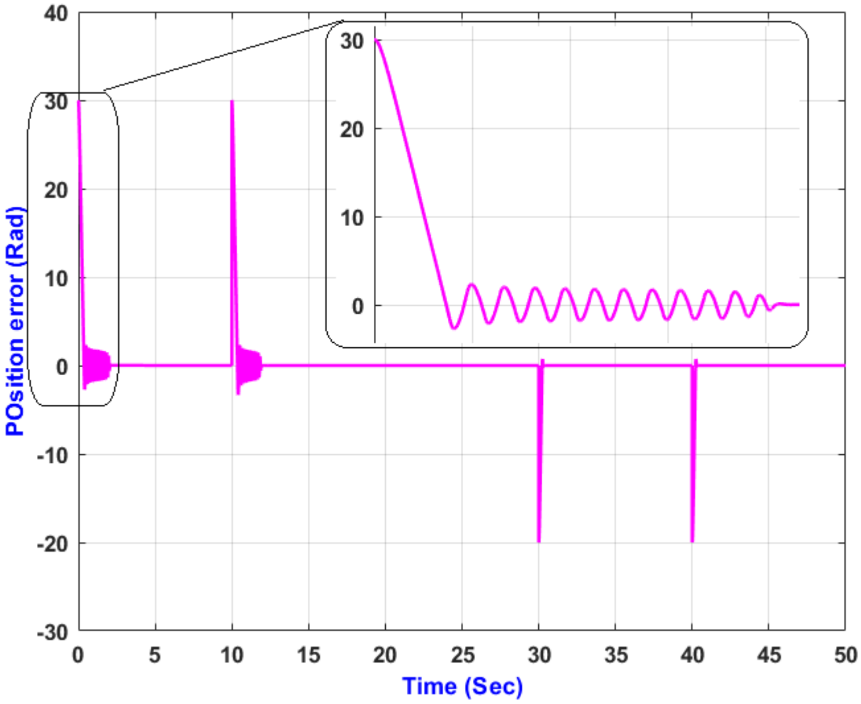


Figure 4.2 Position error result based on CM method at no load

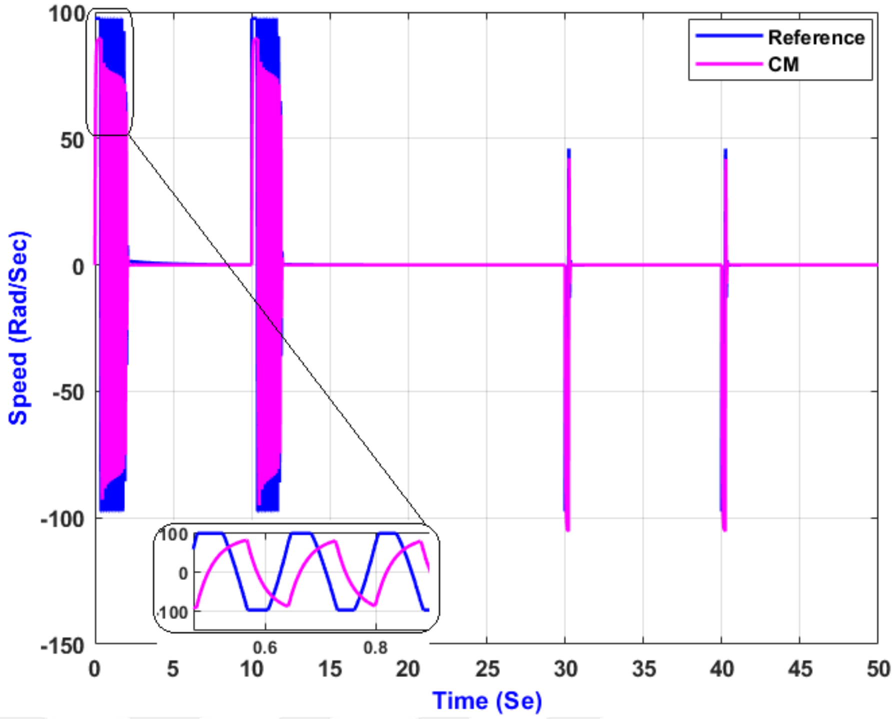


Figure 4.3 Speed result based on CM method at no load

According to the equations of this method referred to in chapter three, Table 4.1 represents the cascade P-PI parameters extracted by the classical method.

Table 4.1 Cascade P-PI parameters based on CM

Cascade P- PI parameters	Value
K_P position	62.8319
K_P speed	2.0944
K_I speed	0.9424
K_P current	0.3817
K_I current	21153

4.1.2 Position and speed results based on genetic algorithm

Figure 4.4 shows the simulation result for the position of the PMDC motor depending on the parameters extracted by the GA algorithm. A large deviation of the position is observed about 27 degrees from the reference position as indicated in Figure 4.5. In addition, it is noted that both the reference and the actual speed of the motor are not uniform as shown in Figure 4.6, because the motor must stop rotating after it reaches the desired position. Here we note that the motor rotates in the opposite direction before it stops after reaching the desired position.

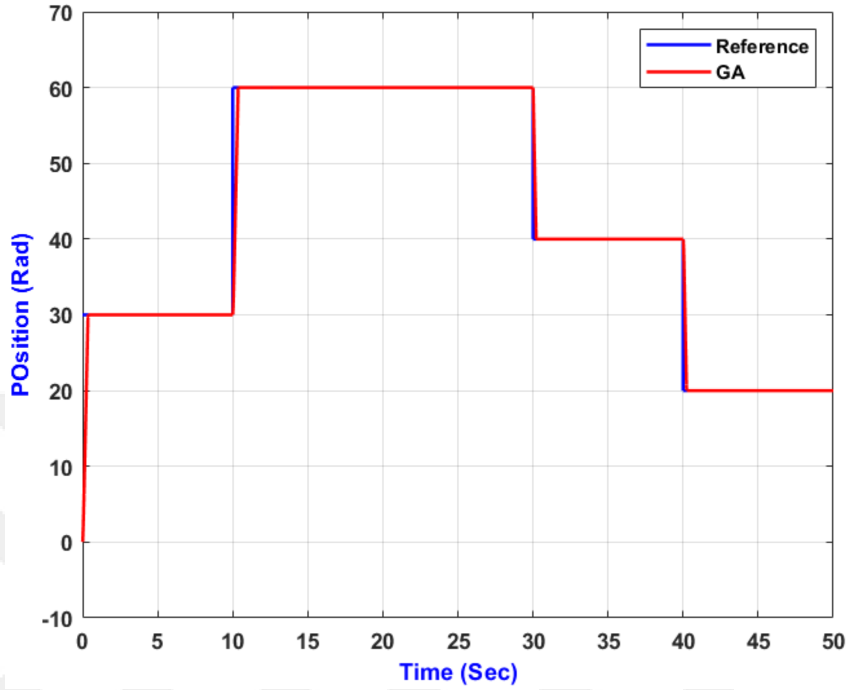


Figure 4.4 Position result based on GA method at 17 Nm load

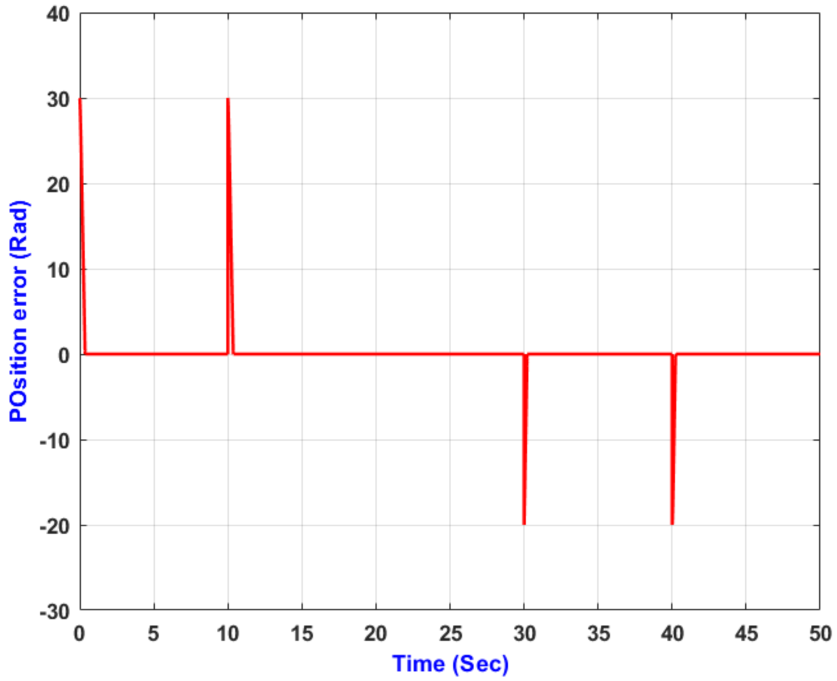


Figure 4.5 Position error result based on GA method at 17 Nm load

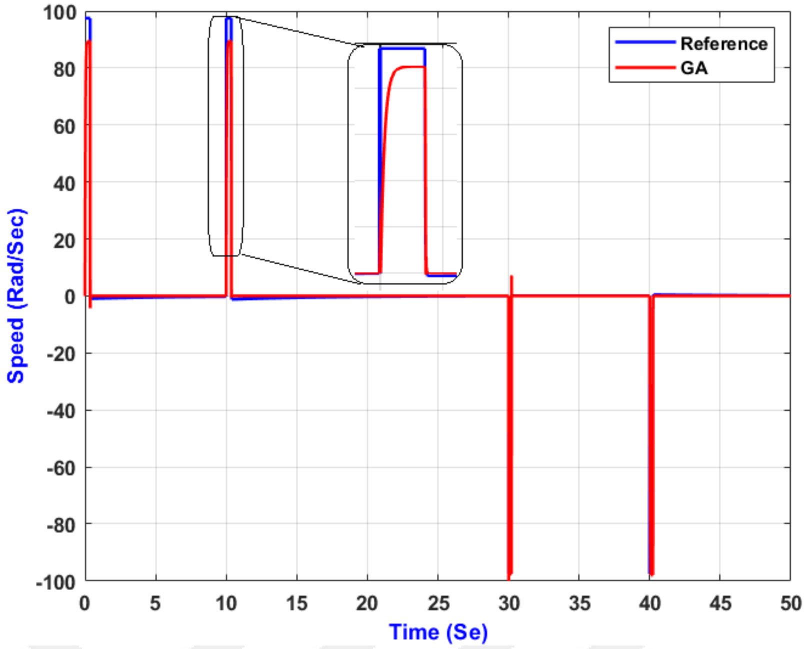


Figure 4.6 Speed result based on GA method at 17 Nm load

Table 4.2 represents the cascade P-PI parameters extracted by a genetic algorithm

Table 4.2 Cascade P-PI parameters based on GA

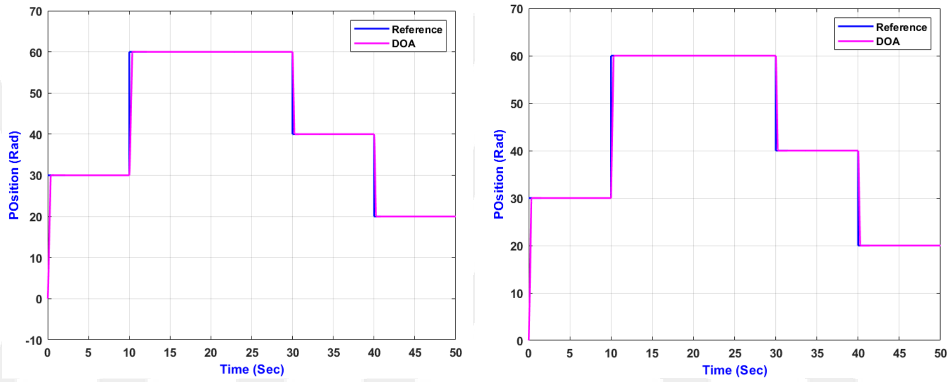
Cascade P- PI parameters	**Value**
KP position	186.4578
KP speed	133.669
KI speed	11.887
KP current	55.6998
KI current	2.365

4.1.3 Position and speed results based on DOA algorithm

The simulation results for PMDC motor position and position error as a function of parameters recovered by the DOA technique are shown in Figures 5.7 and 5.8. A tiny deviation of the position is observed about 1 degree from the reference position. In addition, both the reference and the actual speed of the PMDC motor are uniform as shown in Figure 5.9, the motor must stop rotating after it reaches the desired position. Here we note that the motor not rotates in the opposite direction before it stops after reaching the desired position. The DOA algorithm's retrieved cascade P-PI parameters are shown in Table 4.3.

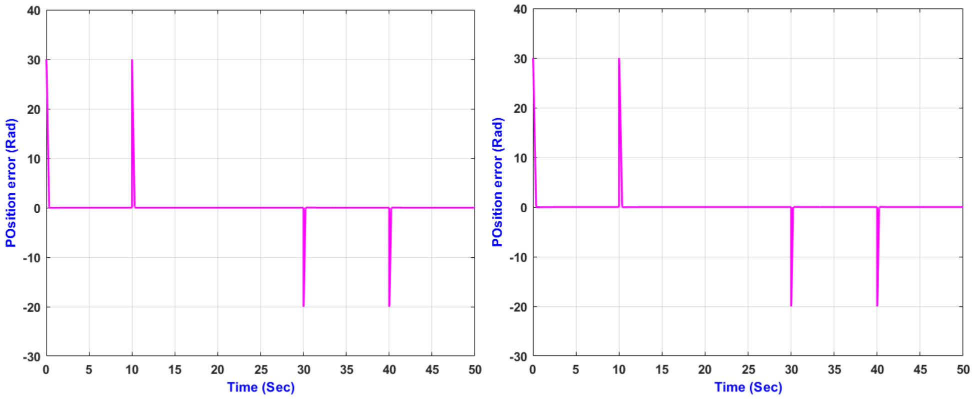
Table 4.3 Cascade P-PI parameters based on DOA

Cascade P- PI parameters	Value
KP position	44.0937
KP speed	63.3902
KI speed	19.4847
KP current	70.639
KI current	3.21979



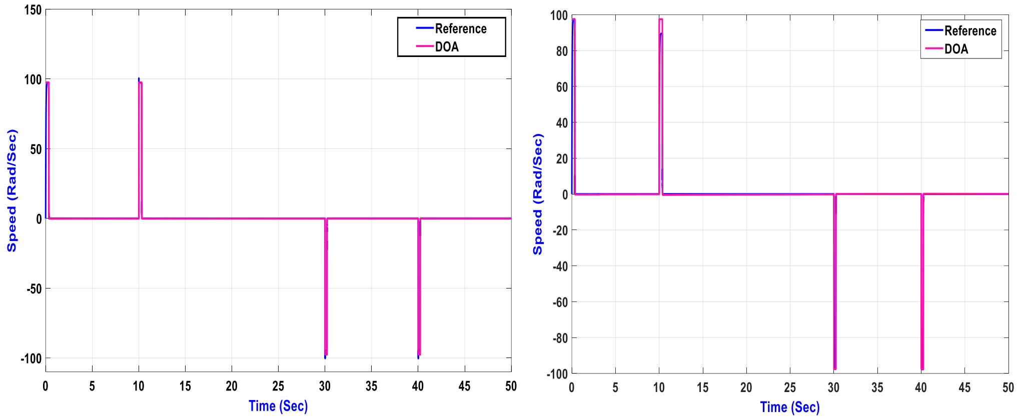
(a) (b)

Figure 4.7 Position result based on DOA algorithm at at (a) no load, (b) load 17 Nm



(a) (b)

Figure 4.8 Position error result based on DOA algorithm at at (a) no load, (b) load 17 Nm



(a) (b)

Figure 4.9 Speed result based on DOA algorithm at (a) no load, (b) load 17 Nm

4.1.4 Position and speed results based on BOA algorithm

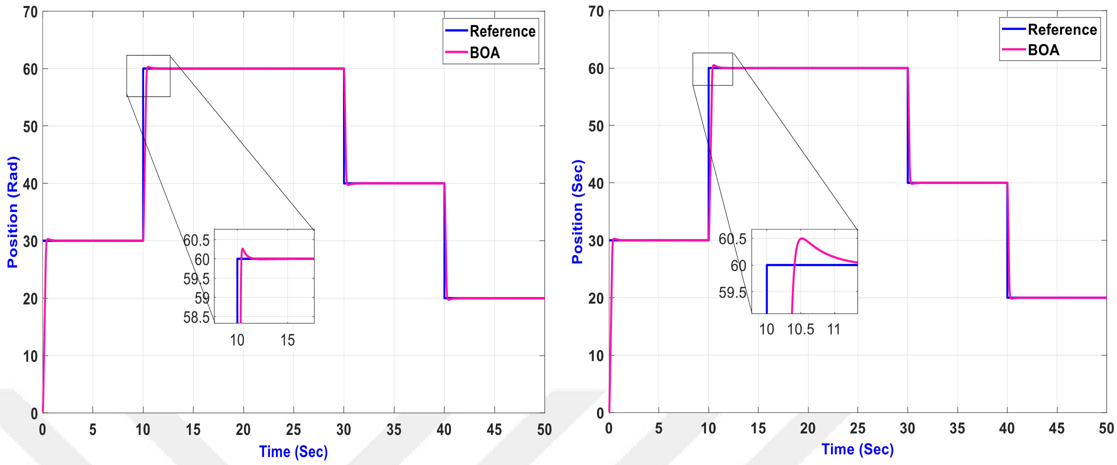
The position and speed of the PMDC motor are controlled using the Butterfly Optimization Algorithm, after the CM, and GA approaches failed to yield satisfactory results. The subsequent results show that this algorithm effectively controls both the speed and position of the PMDC motor, accurately tracking the trajectory of movement along the axis. The BOA method yielded the cascade P-PI parameters shown in Table 4.4.

Table 4.4 Cascade P-PI parameters based on BOA

Cascade P- PI parameters	**Value**
KP position	24.2032
KP speed	26.985
KI speed	38.9883
KP current	8.79824
KI current	46.0113

1) Position simulation results

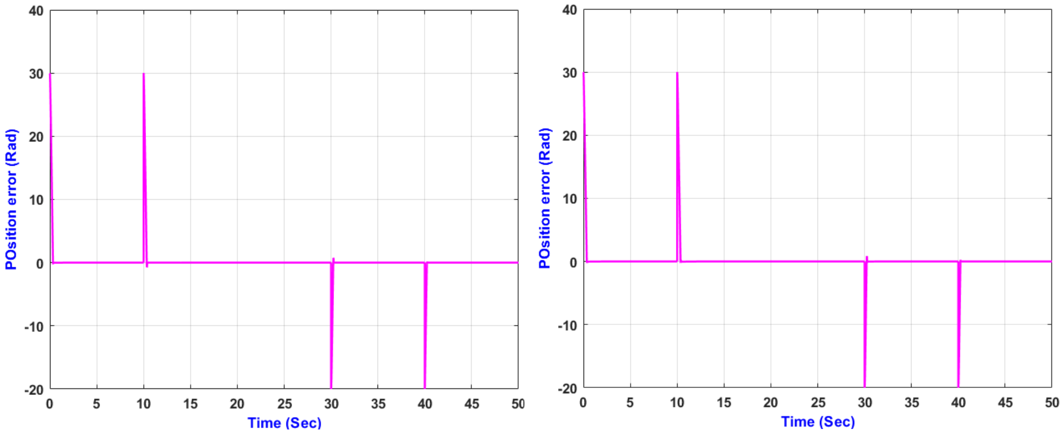
The simulation result for the PMDC motor's location is shown in Figure 4.10 after the BOA approach was used to extract the cascade controller's parameters. The PMDC achieved the target position quickly and accurately under both no-load and load conditions, with no discernible lag or overshoot.



(a) (b)

Figure 4.10 Position result based on BOA algorithm at at (a) no load, (b) load 17 Nm

Figure 4.11 shows that after 50 seconds of operation with full load, the PMDC shows no variation from its real location, demonstrating the durability of the cascade P-PI controller in ignoring disturbances.



(a) (b)

Figure 4.11 Position error result based on BOA algorithm at at (a) no load, (b) load 17 Nm

Using several, high-valued reference places, we can ensure that the PMDC motor will spin for many cycles before stopping where we want it to. No overshoot or divergence from the desired location is shown in Figure 4.10 simulation results, demonstrating the resilience of the cascade P-PI controller in the face of disturbances.

2) Speed simulation results

The simulation results for the PMDC motor's speed under loaded and unloaded conditions are depicted in Figures 4.12 (a) and (b). There is no speed overrun or undershoot, and the motor accurately follows the reference speed trajectory. Upon reaching the set point, the motor halts and then turns to proceed to the next setting. A negative speed value indicates that the motor is descending from a higher position, resulting in a counterclockwise rotation.

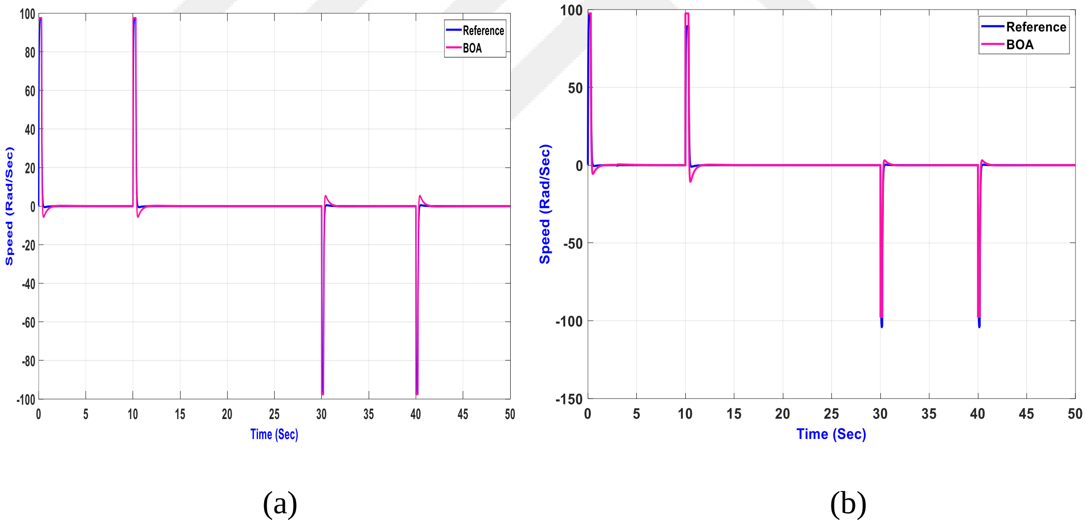


Figure 4.12 Speed result based on BOA algorithm at (a) no load, (b) load 17 Nm

4.2 Comparison between the Results of the Four Tuning Methods

In this thesis, we compare four tuning approaches (CM, GA, BOA and DOA) to determine which one yields the best results for managing the speed and position of the single-axis PMDC.

- The classical method (CM) did not give satisfactory results, as it is observed that the system is unstable when using the parameters extracted by this method, as shown in Figures 4.13 to 4.16.

- As can be seen in Figure 4.13, when the parameters recovered by the GA are used, there is a variation of around 27 degrees from the reference location. As can be seen in Figure 5.14, the speed is erratic, and the motor turns in the wrong direction before coming to stop.

- Figure 4.13 demonstrates that when the parameters recovered by the BOA method are used, there is a variation of around 18 degrees from the reference location. As can be seen in Figure 4.14, the speed is erratic, and the motor turns in the wrong direction before coming to stop.

- The DOA method does not exceed its target velocity or deviate from its reference location, as shown by its simulation results. In addition, the system is evaluated under varied conditions, including with and without load, and with a single or multiple reference positions, and a steady tracking trajectory is shown to arrive at the target location. This is seen in Figures 4.13–4.16. Figures 4.13 and 4.14 indicate that neither the voltage nor the current went above the motor's safe operating ranges, which would have resulted in damage to the device.

Finally, Table 4.5 represents the performance criteria values for position simulation results in full load condition for these three optimal algorithms (BOA, DOA and GA).

Table 4.5 Performance criteria

Performance criteria	**GA**	**DOA**	**BOA**
Rising time (sec)	0.35	0.076	0.08
Settling time (sec)	0.5	0.4	0.4
Overshoot %	8.1128	3.13	4

Therefore, the dandelions optimizer algorithm outperforms the other three tuning strategies by extracting the optimal settings for the cascade P-PI controller, allowing the PMDC motor to consistently and reliably rotate to the target position.

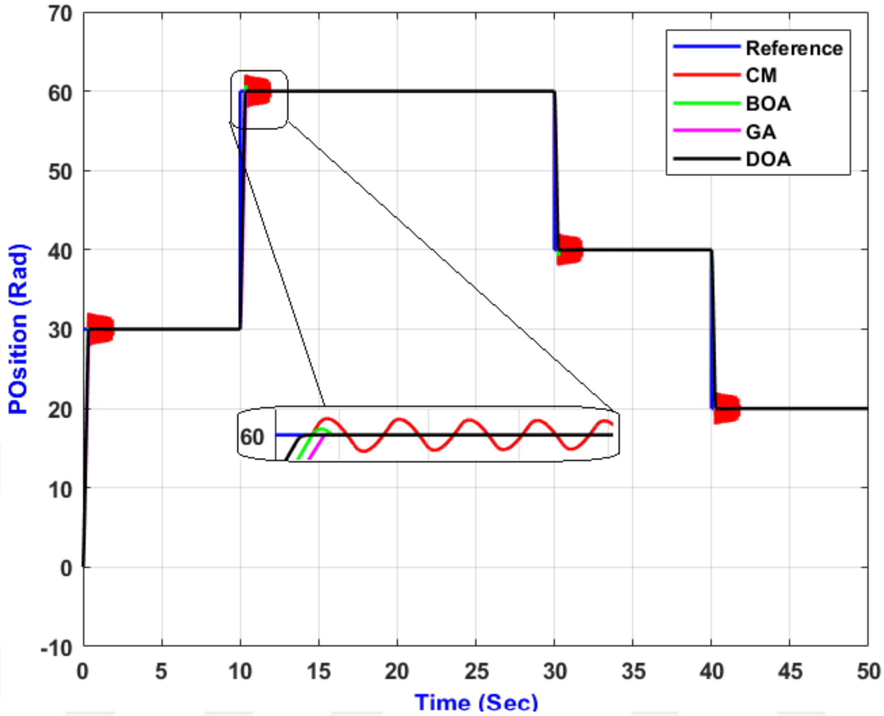


Figure 4.13 Position result based on CM, BOA, GA, and DOA strategies at 17 Nm as load

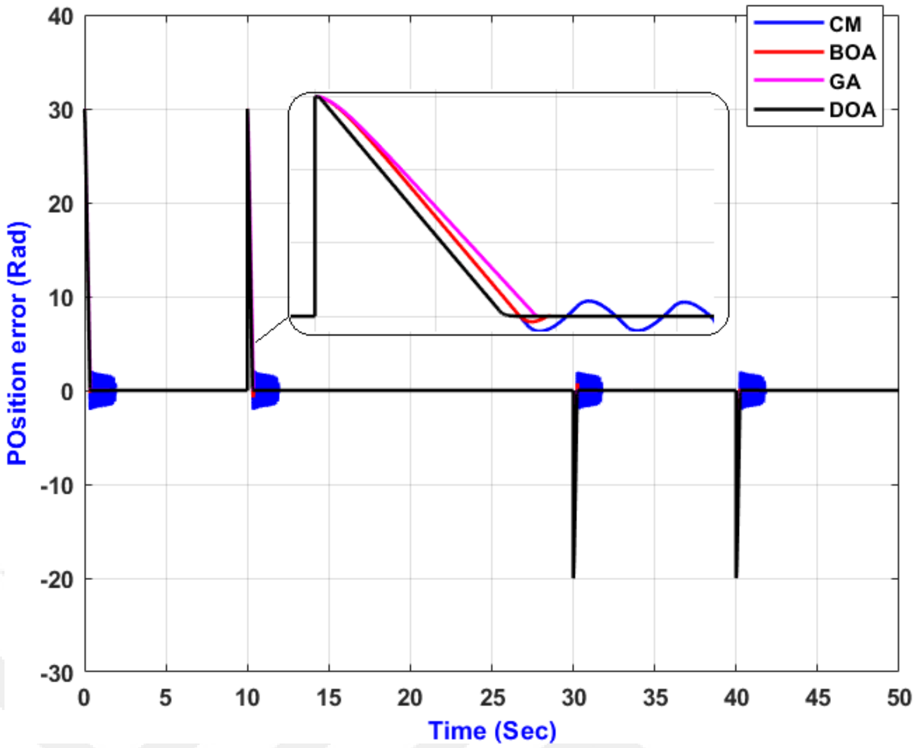


Figure 4.14 Position error result based on CM, BOA, GA, and DOA strategies at 17 Nm as load

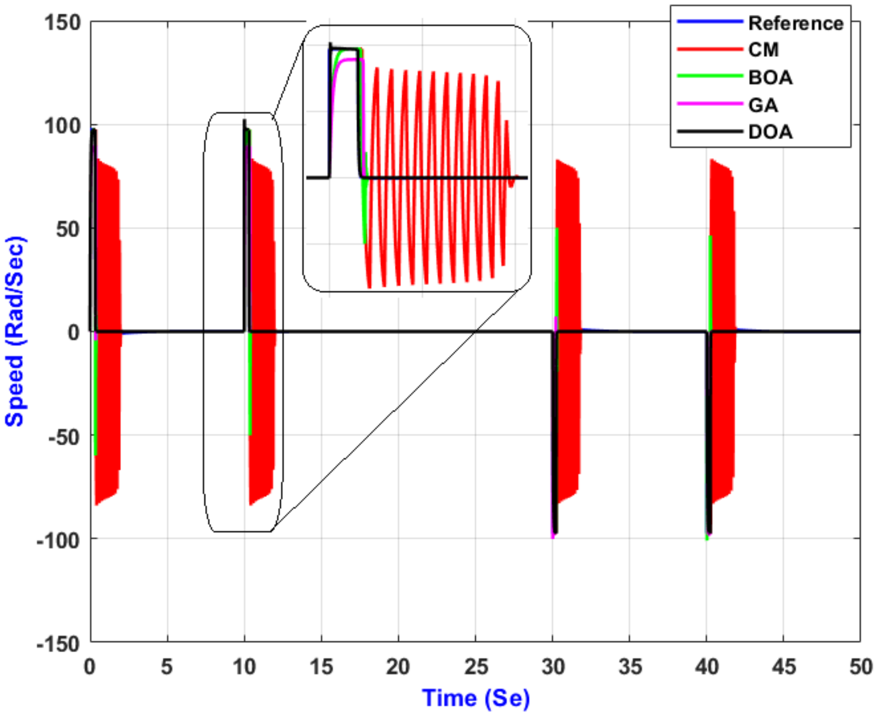


Figure 4.15 Speed result based on CM, BOA, GA, and DOA strategies at 17 Nm as load

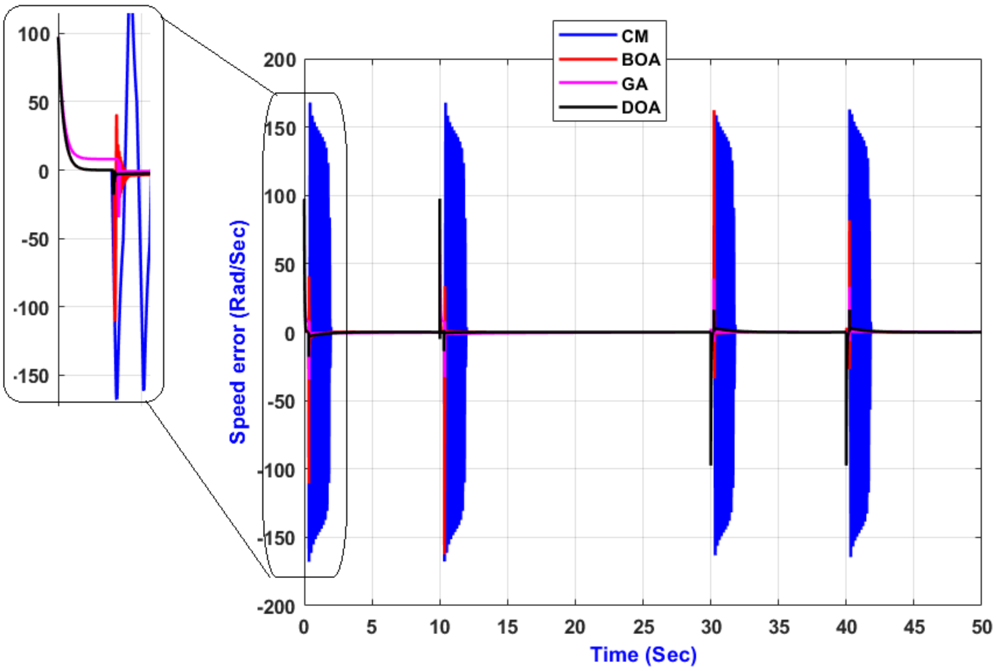


Figure 4.16 Speed error result based on CM, BOA, GA, and DOA strategies at 17 Nm as load

5. CONCLUSION AND RECOMMANDATIONS

In this study, the efficacy and precision of different optimization techniques were assessed for deriving the parameters of the proposed cascade P-PI controller. Among the analyzed methods, DOA finds the best parameter values, ensuring the system achieves the minimum error and the highest success rate. Even though BOA's performance was commendable and closely followed DOA's attributes, GA lagged behind in several performance metrics. The selection of DOA was reinforced by a comparative evaluation, with the ITAE serving as the benchmark performance index, ensuring minimized discrepancies between anticipated and realized positions. The cascade P-PI controllers' resilience was corroborated through a gamut of tests, emphasizing their efficiency in both no-load and full-load scenarios. Under no-load conditions, the theoretical and simulation results mirrored each other closely, ensuring accurate tracking trajectories and desired position attainment. Remarkably, the system's stability was unshaken, and the desired position was achieved even when subjected to a full load, underscoring the robustness of the employed technique.

Based on the findings of this study, it is strongly recommended to prioritize the use of the DOA for applications aiming at optimal system response, given its demonstrated superiority in rising time and minimal overshoot. While the close performance of BOA warrants further exploration, users should exercise caution with the GA due to its lagging metrics. For consistent evaluations, the utilization of ITAE as a benchmark performance index should be continued. Furthermore, the robustness exhibited by the cascade P-PI controllers in varied load conditions suggests their potential applicability in scenarios demanding high stability, but regular reviews are advised to stay abreast with evolving optimization techniques.

REFERENCES

Ali, W. H., Sadiku, M. N. O. and Abood, S. 2019. Fundamentals of Electric Machines: A Primer with MATLAB. CRC Press.

Arasomwan, M. A. and Adewumi, A. O. 2013. On the performance of linear decreasing inertia weight particle swarm optimization for global optimization. The Scientific World Journal, 2013.

Arora, S. and Singh, S. 2019. Butterfly optimization algorithm: a novel approach for global optimization. Soft Computing, 23(3): 715-734.

Bae, J., Cho, K. and Lee, D. H. 2020. Parallel Position Control Scheme of Permanent Magnet DC Motors with a Low-Resolution Sensor. Proceedings of the 2020 IEEE International Conference on Industrial Technology (pp. 1-6). IEEE.

Bharat, S., Ganguly, A., Chatterjee, R., Basak, B., Sheet, D.K. and Ganguly, A. 2019. A review on tuning methods for PID controller. Asian J. Converg. Technol.

Bigelow, T. A. 2020. Electric Circuits, Systems, and Motors (Ch. 5, p. 148). Springer.

Biswas, A. 2013. Physics-inspired optimization algorithms: a survey. Journal of Optimization.

Cankurtaran, M. F. and Kocamis, A. E. 2019. Sensorless Speed Control of PMDC Motor with Cascade PI Controller. In 2019 International Symposium ELMAR. IEEE.

Chau, K. T. and Wang, Z. 2011. Chaos in electric drive systems: analysis, control and application. John Wiley & Sons.

Chen, W. 2013. Speed tracking and synchronization of a dual-motor system via second order sliding mode control. Mathematical Problems in Engineering, 2013.

Ciucur, V.-V. 2014. Speed position control system for DC motor. In 2014 16th International Conference on Harmonics and Quality of Power (ICHQP). IEEE.

Cuong, N. D. and Puta, H. 2013. Design of MRAS based control systems for load sharing of two DC motors with a common stiff shaft. In 2013 International Conference on Control, Automation and Information Sciences (ICCAIS). IEEE.

Divya, N. and Nirmalkumar, A. 2018. A survey on tuning of PID controller for industrial process using soft computing techniques. Int. J. Pure Appl. Math. 118: 663–667.

Dorf, R. C. and Bishop, R. H. 2011. Modern control systems. Pearson.

El-Sharkawi, M. A. 2015. Electric energy: an introduction. CRC press.

Flores-Morán, E., Yánez-Pazmiño, W. and Barzola-Monteses, J. 2018. Genetic algorithm and fuzzy self-tuning PID for DC motor position controllers. Proceedings of the 2018 19th International Carpathian Control Conference, pp. 1-5. IEEE.

Franchi, C. M. 2019. Electrical Machine Drives: Fundamental Basics and Practice 1(1). CRC Press.

Gholap, V., Dessai, C. N. and Bagyaveereswaran, V. 2017. PID controller tuning using metaheuristic optimization algorithms for benchmark problems. IOP Conference Series: Materials Science and Engineering, 263: 052021.

Guzey, H. M. 2018. Optimal synchronizing speed control of multiple DC motors. Proceedings of the 2018 4th International Conference on Optimization and Applications, pp. 1-5. IEEE.

Gücin, T. N.2015. Tuning cascade PI (D) controllers in PMDC motor drives: A performance comparison for different types of tuning methods. In 2015 9th International Conference on Electrical and Electronics Engineering (ELECO). IEEE.

Hasan, I. J., Ab Ghani, M. R. and Gan, C. K. 2014. Optimum distributed generation allocation using PSO in order to reduce losses and voltage improvement.

Ishizaki, K., Sencer, B. and Shamoto, E. 2013. Cross Coupling Controller for Accurate Motion Synchronization of Dual Servo Systems. IJAT, 7(5): 514-522.

Kang, J. S. 2016. Electric Circuits (Ch. 1, p. 11). Cengage Learning.

Krause, P. C., Wasynczuk, O., Sudhoff, S. D. and Pekarek, S. 2002. Analysis of electric machinery and drive systems, Wiley Online Library, 2(10): 396-397.

Kumar, S. M. G., Jayaraj, D. and Kishan, A. R. 2010. PSO based tuning of a PID controller for a high performance drilling machine. International Journal of Computer Applications, 1(19): 12-18.

Latif, A. 2020. Optimum Synthesis of a BOA optimized novel dual-stage PI-(1+ ID) controller for frequency response of a microgrid. Energies, 13(13): 3446.

Levine, W. S. 2019. Control System Fundamentals. CRC Press.

Mehta, V. K. and Mehta, R. 2008. Principles of electrical machines. S. Chand Publishing.

Mhase, V. 2012. Integrated speed–position tracking with trajectory generation and synchronization for 2–axis DC motion control. International Journal of Engineering Research and Development, 1(6): 61-66.

Mirjalili, S. 2016. SCA: a sine cosine algorithm for solving optimization problems. Knowledge-based systems, 96: 120-133.

Momoh, J. A. 2018. Energy Processing and Smart Grid (Ch. 8, p. 169). John Wiley & Sons.

Morar, A. 2007. DC motor speed and position control system. IFAC Proceedings Volumes, 40(8): 203-208.

Namazov, M. and Basturk, O. 2010. DC motor position control using fuzzy proportional-derivative controllers with different defuzzification methods.

Pinto, V. H., Gonçalves, J. and Costa, P. 2020. Modeling and Control of a DC Motor Coupled to a Non-Rigid Joint. Applied System Innovation, 3(2): 24.

Pisano, A. 2008. Cascade control of PM DC drives via second-order sliding-mode technique. IEEE Transactions on Industrial Electronics, 55(11): 3846-3854.

Raja, G. L. and Ali, A. 2017. Series cascade control: An outline survey. In 2017 Indian Control Conference (ICC). IEEE.

Romero, L. and Concha, A. 2006. Control of position/velocity in a mobile robot using dc brushless motors. In Electronics, Robotics and Automotive Mechanics Conference, 2.

Sinha, N. K. and Tiwari, P. M. 2017. Multiple motor synchronization using nonlinear PID control. Proceedings of the 2017 3rd International Conference on Advances in Computing, Communication & Automation, pp. 1-5. IEEE.

Slowik, A. and Kwasnicka, H. 2020. Evolutionary algorithms and their applications to engineering problems. Neural Computing and Applications.

Son, Y. I. 2014. Robust cascade control of electric motor drives using dual reduced-order PI observer. IEEE Transactions on Industrial Electronics, 62(6): 3672-3682.

Sun, D. and Mills, J. K. 2002. Adaptive synchronized control for coordination of multirobot assembly tasks. IEEE Transactions on Robotics and Automation, 18(4): 498-510.

Tang, J. 2001. PID controller using the TMS320C31 DSK with online parameter adjustment for real-time DC motor speed and position control. In ISIE 2001. 2001 IEEE International Symposium on Industrial Electronics Proceedings, 2.

Taut, M. A. 2018. Model-in-the-Loop for Determining the Speed and Position of a DC Motor. In 2018 41st International Spring Seminar on Electronics Technology (ISSE). IEEE.

Wang, X. and Suh, C. S. 2017. Precision concurrent speed and position tracking of brushed dc motors using nonlinear time-frequency control. Journal of Vibration and Control, 23(19): 3266-3291.

Xiao, Y. 2013. Synchronous control for high-accuracy biaxial motion systems. Journal of Control Theory and Applications, 11(2): 294-298.

Yazgan, H. R. 2019. Comparison performances of PSO and GA to tuning PID controller for the dc motor. Sakarya Üniversitesi Fen Bilimleri Enstitüsü Dergisi, 23(2): 162-174.

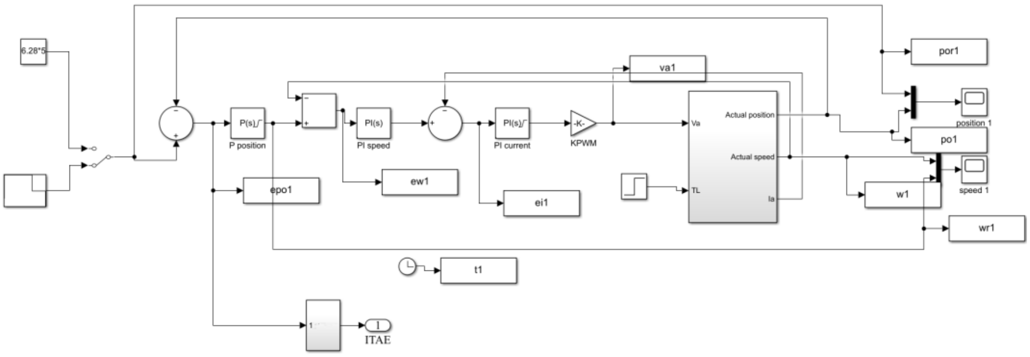
Yeh, S. and Hsu, P. L. 2000. Design of precise multi-axis motion control systems. Proceedings of the 6th International Workshop on Advanced Motion Control, pp. 1-8. IEEE.

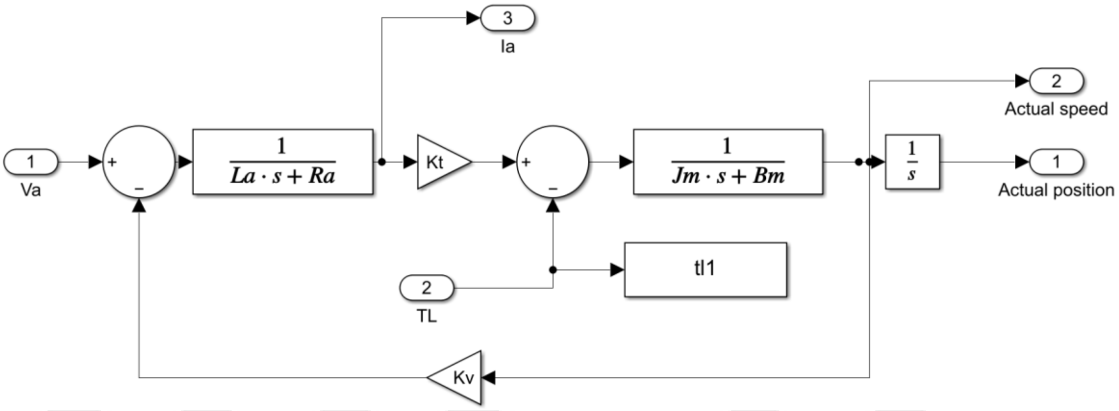
Zhang, P. and Wang, Z. 2019. Improvements of direct current motor control and motion trajectory algorithm development for automated guided vehicle. Advances in Mechanical Engineering, 11(2).

Zhao, S. 2022. Dandelion optimizer: A nature-inspired metaheuristic algorithm for engineering applications. Engineering Applications of Artificial Intelligence, 114: 105075.

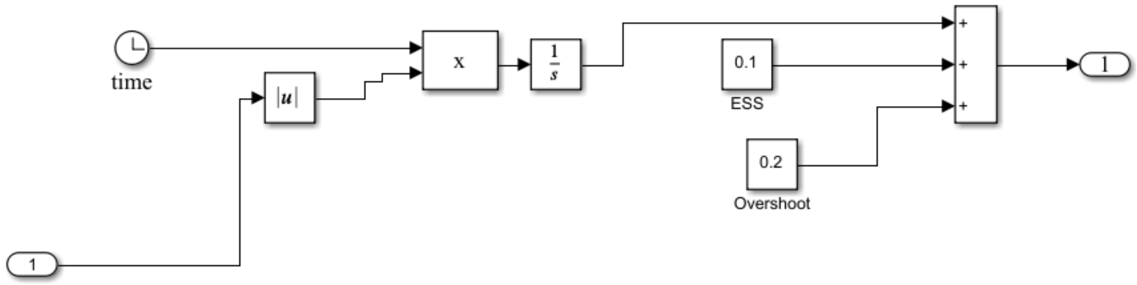
APPENDICES

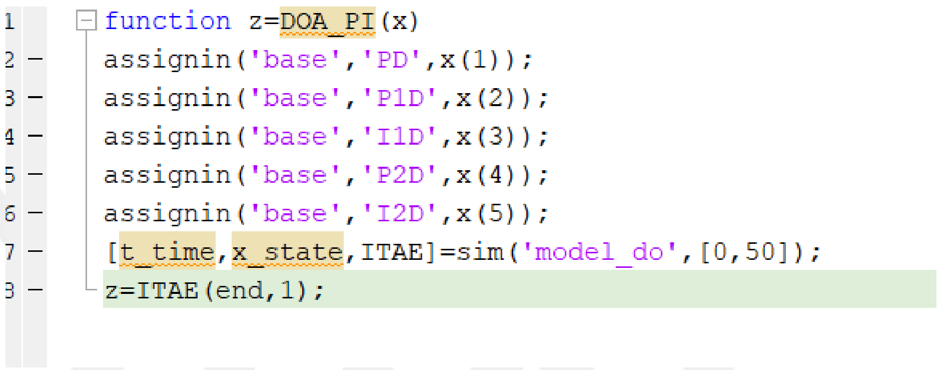
APPENDIX 1. PMDC motor Simulink





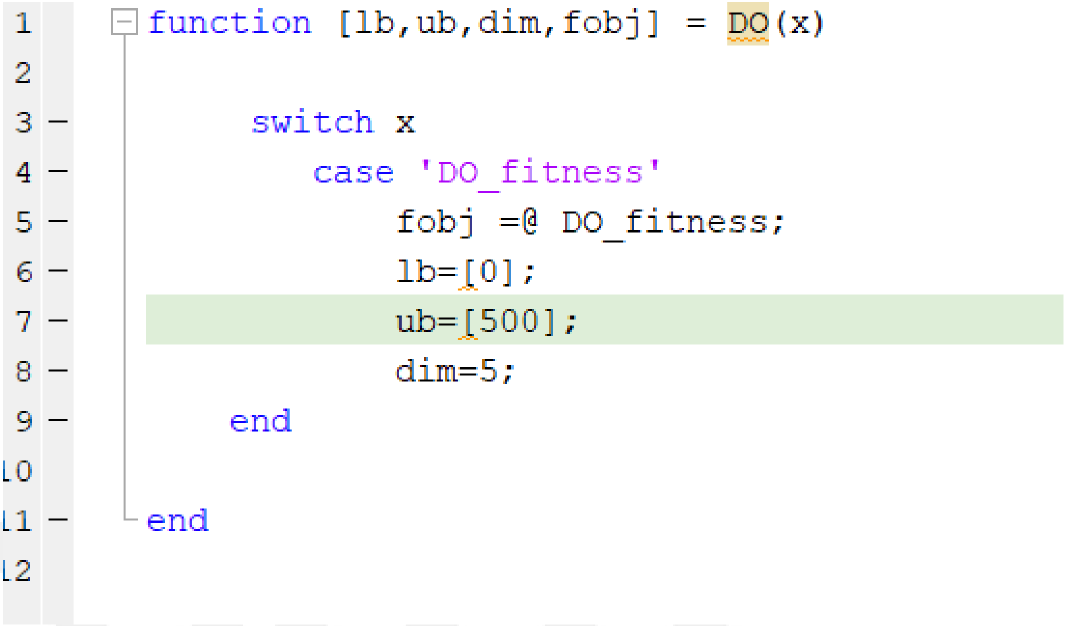
APPENDIX 2. Fitness function equations and m.file code





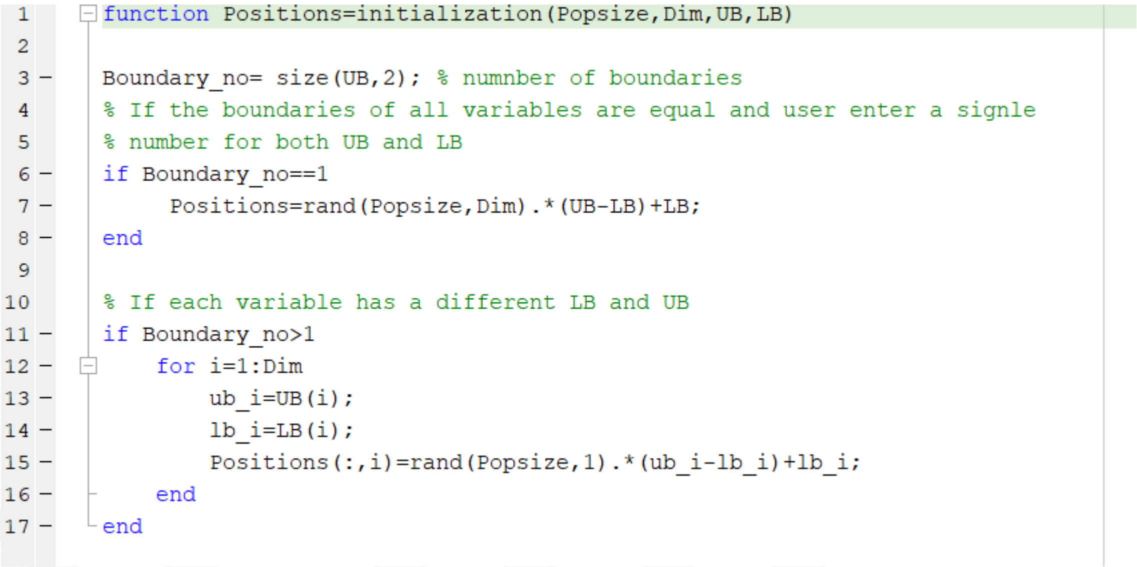
```
function z=DOA_PI(x)
assignin('base','PD',x(1));
assignin('base','P1D',x(2));
assignin('base','I1D',x(3));
assignin('base','P2D',x(4));
assignin('base','I2D',x(5));
[t_time,x_state,ITAE]=sim('model_do',[0,50]);
z=ITAE(end,1);
```

APPENDIX 3. Matlab code of constraints



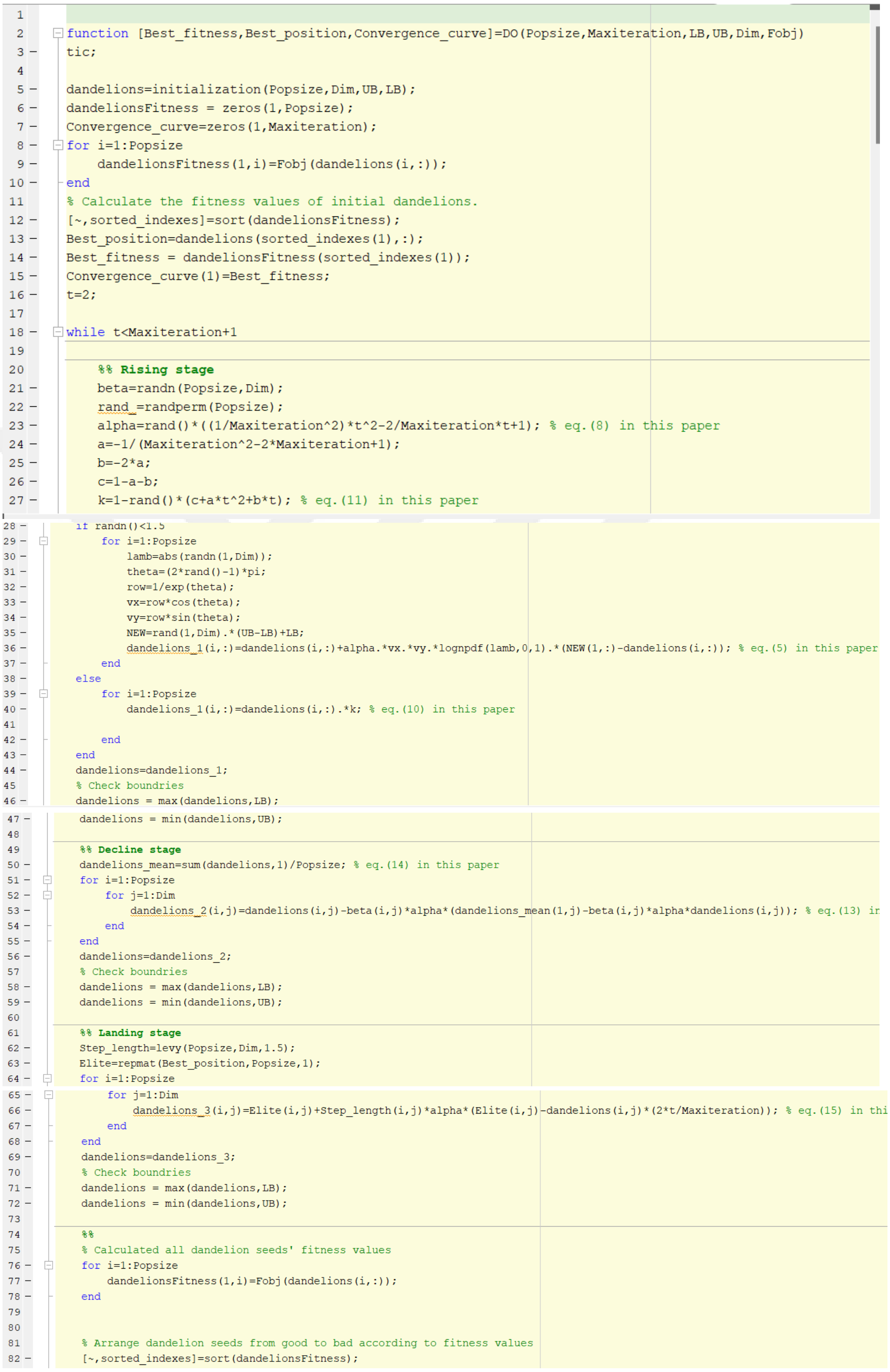
```
function [lb,ub,dim,fobj] = DO(x)

     switch x
        case 'DO_fitness'
            fobj =@ DO_fitness;
            lb=[0];
            ub=[500];
            dim=5;
    end

end
```

APPENDIX 4. Initialize matlab function

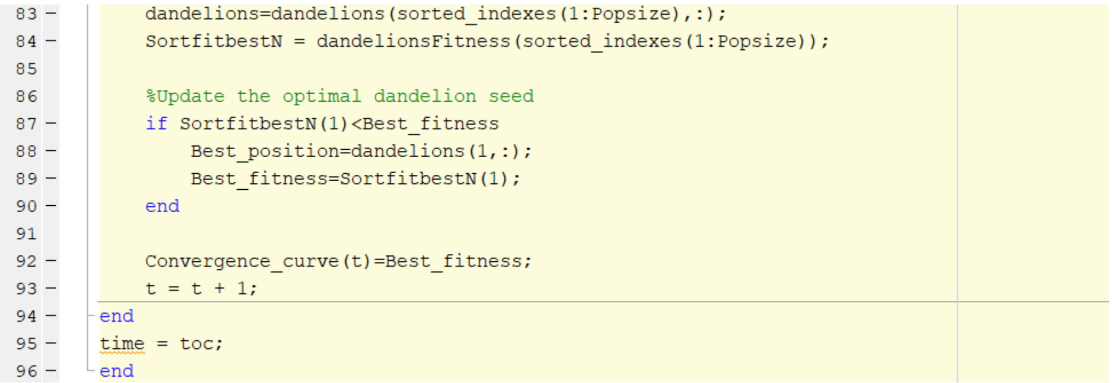


```
function Positions=initialization(Popsize,Dim,UB,LB)

Boundary_no= size(UB,2); % numnber of boundaries
% If the boundaries of all variables are equal and user enter a signle
% number for both UB and LB
if Boundary_no==1
     Positions=rand(Popsize,Dim).*(UB-LB)+LB;
end

% If each variable has a different LB and UB
if Boundary_no>1
    for i=1:Dim
        ub_i=UB(i);
        lb_i=LB(i);
        Positions(:,i)=rand(Popsize,1).*(ub_i-lb_i)+lb_i;
    end
end
```

APPENDIX 5. MATLAB code of DOA



```
function [Best_fitness,Best_position,Convergence_curve]=DO(Popsize,Maxiteration,LB,UB,Dim,Fobj)
tic;

dandelions=initialization(Popsize,Dim,UB,LB);
dandelionsFitness = zeros(1,Popsize);
Convergence_curve=zeros(1,Maxiteration);
for i=1:Popsize
    dandelionsFitness(1,i)=Fobj(dandelions(i,:));
end
% Calculate the fitness values of initial dandelions.
[~,sorted_indexes]=sort(dandelionsFitness);
Best_position=dandelions(sorted_indexes(1),:);
Best_fitness = dandelionsFitness(sorted_indexes(1));
Convergence_curve(1)=Best_fitness;
t=2;

while t<Maxiteration+1

    %% Rising stage
    beta=randn(Popsize,Dim);
    rand_=randperm(Popsize);
    alpha=rand()*((1/Maxiteration^2)*t^2-2/Maxiteration*t+1); % eq.(8) in this paper
    a=-1/(Maxiteration^2-2*Maxiteration+1);
    b=-2*a;
    c=1-a-b;
    k=1-rand()*(c+a*t^2+b*t); % eq.(11) in this paper
    if randn()<1.5
        for i=1:Popsize
            lamb=abs(randn(1,Dim));
            theta=(2*rand()-1)*pi;
            row=1/exp(theta);
            vx=row*cos(theta);
            vy=row*sin(theta);
            NEW=rand(1,Dim).*(UB-LB)+LB;
            dandelions_1(i,:)=dandelions(i,:)+alpha.*vx.*vy.*lognpdf(lamb,0,1).*(NEW(1,:)-dandelions(i,:)); % eq.(5) in this paper
        end
    else
        for i=1:Popsize
            dandelions_1(i,:)=dandelions(i,:).*k; % eq.(10) in this paper

        end
    end
    dandelions=dandelions_1;
    % Check boundries
    dandelions = max(dandelions,LB);
    dandelions = min(dandelions,UB);

    %% Decline stage
    dandelions_mean=sum(dandelions,1)/Popsize; % eq.(14) in this paper
    for i=1:Popsize
        for j=1:Dim
            dandelions_2(i,j)=dandelions(i,j)-beta(i,j)*alpha*(dandelions_mean(1,j)-beta(i,j)*alpha*dandelions(i,j)); % eq.(13) i
        end
    end
    dandelions=dandelions_2;
    % Check boundries
    dandelions = max(dandelions,LB);
    dandelions = min(dandelions,UB);

    %% Landing stage
    Step_length=levy(Popsize,Dim,1.5);
    Elite=repmat(Best_position,Popsize,1);
    for i=1:Popsize
        for j=1:Dim
            dandelions_3(i,j)=Elite(i,j)+Step_length(i,j)*alpha*(Elite(i,j)-dandelions(i,j)*(2*t/Maxiteration)); % eq.(15) in thi
        end
    end
    dandelions=dandelions_3;
    % Check boundries
    dandelions = max(dandelions,LB);
    dandelions = min(dandelions,UB);

    %%
    % Calculated all dandelion seeds' fitness values
    for i=1:Popsize
        dandelionsFitness(1,i)=Fobj(dandelions(i,:));
    end


    % Arrange dandelion seeds from good to bad according to fitness values
    [~,sorted_indexes]=sort(dandelionsFitness);
```



```
    dandelions=dandelions(sorted_indexes(1:Popsize),:);
    SortfitbestN = dandelionsFitness(sorted_indexes(1:Popsize));

    %Update the optimal dandelion seed
    if SortfitbestN(1)<Best_fitness
        Best_position=dandelions(1,:);
        Best_fitness=SortfitbestN(1);
    end

    Convergence_curve(t)=Best_fitness;
    t = t + 1;
end
time = toc;
end
```

CURRICULUM VITAE

Personal Information

Name and Surname : Khalaf Abdullah Khalaf KHALAF

Education

MSc	Çankırı Karatekin University Graduate School of Natural and Applied Sciences Department of Electrical and Electronics Engineering	2021-2023
Undergraduate	University of Tikrit College of Engineering Department of Electrical Engineering	2012-2016