REAL TIME CALCULATION OF SHEM ANGLES BY METAHEURISTIC ALGORITHMS FOR VARIABLE FREQUENCY MOTOR DRIVES

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY

GAMZE NALÇACI

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF DOCTOR OF PHILOSOPHY
IN
ELECTRICAL AND ELECTRONIC ENGINEERING

SEPTEMBER 2021

Approval of the thesis:

REAL TIME CALCULATION OF SHEM ANGLES BY METAHEURISTIC ALGORITHMS FOR VARIABLE FREQUENCY MOTOR DRIVES

submitted by **GAMZE NALÇACI** in partial fulfillment of the requirements for the degree of **Doctor of Philosophy in Electrical and Electronic Engineering, Middle East Technical University** by,

Prof. Dr. Halil Kalıpçılar
Dean, Graduate School of **Natural and Applied Sciences** _____

Prof. Dr. İlkay Ulusoy
Head of the Department, **Electrical and Electronics Eng.** _____

Prof. Dr. Muammer Ermiş
Supervisor, **Electrical and Electronics Eng., METU** _____

Prof. Dr. Işık Çadırcı
Co-Supervisor, **Electrical and Electronics Eng., HU**
_____

Examining Committee Members:

Assoc. Prof. Dr. Ozan Keysan
Electrical and Electronics Eng, METU _____

Prof. Dr. Muammer Ermiş
Electrical and Electronics Eng, METU _____

Assist. Prof. Dr. Emine Bostancı Özkan
Electrical and Electronics Eng, METU _____

Prof. Dr. Mehmet Timur Aydemir
Electrical and Electronics Eng, Kadir Has University _____

Assist. Prof. Dr. Mümtaz Mutluer
Electrical and Electronics Eng, Necmettin Erbakan University _____

Date: 23.09.2021

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.

Name, Last name : Gamze Nalçacı

Signature :

ABSTRACT

REAL TIME CALCULATION OF SHEM ANGLES BY METAHEURISTIC ALGORITHMS FOR VARIABLE FREQUENCY MOTOR DRIVES

Nalçacı, Gamze
Doctor of Philosophy, Electrical and Electronic Engineering
Supervisor: Prof. Dr. Muammer Ermiş
Co-Supervisor: Prof. Dr. Işık Çadırcı

September 2021, 158 pages

Harris hawks optimization (HHO) has been used to optimize a voltage source inverter's selective harmonic elimination method (SHEM) for light rail transportation. The primary goal of SHEM's pulse-width modulation strategy is to eliminate low-order harmonics by solving nonlinear equations while maintaining the required magnitude of the fundamental component. When applied to a two-level, three-phase inverter, the HHO algorithm aims to solve the associated nonlinear equations with a high degree of accuracy and probability of convergence, outperforming particle swarm optimization (PSO), grey wolf optimization (GWO), and whale optimization algorithms (WOA).

SHEM simulations using MATLAB / Simulink are performed on a 125 kW traction motor drive platform for various SHEM angles in order to compare its performance to SHEM using SPWM and SVM techniques. Experiments conducted on a laboratory-scale physical simulator of a light rail transportation system revealed remarkable findings regarding the effect of the DC link voltage magnitude on the performance of the associated traction motor drive. In light of the various fundamental frequency operations, and the operating DC link voltages, a general

assessment of the SHEM has been made in light of its application to a light rail traction motor drive. Additionally, the calculated SHEM switching angles using PSO, WOA, GWO, and HHO in compute unified device architecture (CUDA) are compared on two graphics processing units (GPUs), with the results indicating that HHO calculates faster than the other metaheuristic methods for real-time applications.

Keywords: Real time calculation, metaheuristic algorithms, motor drives, selective harmonic elimination, graphic processing unit

ÖZ

DEĞİŞKEN FREKANSLI MOTOR SÜRÜCÜLERİ İÇİN METASEZGİSEL ALGORİTMALAR KULLANILARAK SHEM AÇILARININ GERÇEK ZAMANLI HESAPLANMASI

Nalçacı, Gamze
Doktora, Elektrik ve Elektronik Mühendisliği
Tez Yöneticisi: Prof. Dr. Muammer Ermiş
Ortak Tez Yöneticisi: Prof. Dr. Işık Çadırcı

Eylül 2021, 158 sayfa

Harris hawks optimizasyonu (HHO), hafif raylı ulaşım için bir voltaj kaynağı invertörünün seçici harmonik eliminasyon yöntemini (SHEM) optimize etmek için kullanılmıştır. SHEM'in darbe genişliği modülasyon stratejisinin birincil amacı, temel bileşenin gerekli büyüklüğünü korurken doğrusal olmayan denklemleri çözerek düşük dereceli harmonikleri ortadan kaldırmaktır. İki seviyeli, üç fazlı bir invertöre uygulandığında, parçacık sürü optimizasyonu (PSO), gri kurt optimizasyonu (GWO) ve balina optimizasyon algoritmaları (WOA) dan daha iyi performans gösteren HHO algoritması ilişkili doğrusal olmayan denklemleri yüksek derecede doğruluk ve yakınsama olasılığı ile çözmeyi amaçlamaktadır.

MATLAB / Simulink simülasyonları kullanılarak, SPWM ve SVM tekniklerinin performansını SHEM ile karşılaştırmak için çeşitli SHEM açılarının testleri 125 kW'lık bir cer motoru sürücü platformunda gerçekleştirilmektedir. Bir hafif raylı ulaşım sisteminin laboratuvar ölçekli bir fiziksel simülatörü üzerinde yapılan deneyler, DC bağlantı voltajının büyüklüğünün ilgili cer motoru sürücüsünün performansı üzerindeki etkisine ilişkin dikkate değer bulgular ortaya çıkarmıştır.

Çeşitli temel frekans işlemleri ve çalışma DC bağlantı voltajları ışığında, hafif raylı bir cer motoru sürücüsüne uygulanması ışığında SHEM'in genel bir değerlendirmesi yapılmıştır. Ek olarak, hesaplama birleşik cihaz mimarisinde (CUDA) PSO, WOA, GWO ve HHO kullanılarak hesaplanan SHEM anahtarlama açıları, iki grafik işleme biriminde (GPU'lar) karşılaştırılır ve sonuçlar HHO'nun gerçek-zamanlı uygulamalar için diğer meta-sezgisel yöntemlerden daha hızlı hesaplama yaptığını göstermektedir.

Anahtar Kelimeler: Gerçek zamanlı hesaplama, metasezgisel algoritmalar, motor sürücüler, seçmeli harmonik eliminasyonu, grafik işleme ünitesi

To My Family

ACKNOWLEDGMENTS

First of all, I would like to express my deepest gratitude to my advisors, Prof. Dr. Muammer Ermiş and Prof. Dr. Işık Çadırcı, for their guidance throughout the research process, encouragement and supervision, and their valuable contributions to me. I am also very grateful for the helpful comments and suggestions of my thesis committee.

I would also thank Doğan Yıldırım for his valuable comments, suggestions, encouragements, and support for the experimental setup.

Thank my friends for their sacrifice, unconditional support, and encouragement. Last but not least, I would like to thank my beloved family. Without their love, motivation, and support, I cannot finish this thesis.

This work is partially funded by the Scientific and Technological Research Council of Turkey under grant number TUBİTAK 112E506.

TABLE OF CONTENTS

LIST OF TABLES

TABLES

LIST OF FIGURES

FIGURES

LIST OF ABBREVIATIONS

ABBREVIATIONS	
AAC	Asynchronous Alternating Current
AC	Alternating Current
AI	Artificial Intelligence
AM	Arithmetic Method
ANN	Artificial Neural Network
BESS	Battery Energy Storage System
BFA	Bacterial Foreaging Algorithm
BIA	Bio-Intelligent Algorithms
BOA	Bat Optimization Algorithm
CPU	Central Processing Unit
CSA	Cuckoo Search Algorithm
CUDA	Compute Unified Device Architecture
DC	Direct Current
DE	Differential Evoluation
DSP	Digital Signal Processor
FA	Firefly Algorithm
FAGA	Firefly Assisted Genetic Algorithm
FPGA	Field-programmable Gate Array
GA	Genetic Algorithm

GPU	Graphic Processing Unit
GPGPU	General Purpose Graphic Processing Unit
GWO	Grey Wolf Optimizer
HE	Harmonic Elimination
HPC	High-performance Computing
HVDC	High Voltage Direct Current
IGCT	Integrated Gate Commutated Thyristors
HRCGA	High-rate Coded Genetic Algorithm
HHO	Harris Hawks Optimization
IC	Incremental Conductivity
ICA	Imperialist Competitive Algorithm
IGBT	Insulated-gate Bipolar Transistor
IVA	Intelligent Video Analytics
LRT	Light Rail Transit
MCT	MOS Contralled Thyristors
MOSFET	Metal-oxide Semiconductor Field Effect Transistor
MPCore	Quad-Core
NM	Numeric Method
NR	Newton Raphson
NVCC	NVIDIA C Compiler
PV	Photo-voltaic
PWM	Pulse Width Modulation

PSO	Particle Swarm Optimization
SFLA	Shuffled Frog Leaping Algorithm
SiC	Silicon Carbide
SHE-PWM	Selective Harmonic Elimination Pulse Width Modulation
SHEM	Selective Harmonic Elimination Method
SM	Streaming Multiprocessor
SoM	System on Module
SPWM	Sinusoidal Pulse Width Modulation
SVPWM	Space Vector Pulse Width Modulation
VFD	Variable Frequency Drive
VR	Virtually Reality
VSC	Voltage Source Converter
VSI	Voltage Source Inverter
2D	Two-dimendional
3D	Three-dimensional
TFLOPS	Terraflops Per Second
TLBO	Teaching-learning Based Optimizer
THD	Total Harmonic Distortion
TLL	Technique of Line to Line
TLN	Technique of Line to Neutral
UPS	Uniterruptable Power Supply
WOA	Whale Optimizer Algorthm

LIST OF SYMBOLS

SYMBOLS

α_k	Switching Angles
m	Modulation Index
v_{id}	Velocity of particles
x_{id}	Position of Particles
v_{max}	Maximum Velocity
c_1	Cognition Factor
c_2	Social Factor
p_{id}	Neighbourhood Best Position
t	The Number of Iteration
A	Coefficient Vector
D	Coefficient Vector
a	Random Number
C	Prey Interval
α	Alfa Wolf Coefficient
β	Beta Wolf Coefficient
δ	Delta Wolf Coefficient
X	Position Vector of Best Solution
r	Random Vector
X_{rand}	Randomly Generated Position Vector

X_i	Position Vector of Hawks
X_{prey}	Position Vector of Rabbit
ub	Upper Boundary of Best Solutions
lb	Lower Boundary of Best Solutions
X_m	Each Hawk's Location
$levy$	Levy Flight Model
f	Fitness Value
V_{no}	Number of Model Parameters
w	System Frequeny
n	Harmonic Number
f_{cost}	Cost Function

CHAPTER 1

INTRODUCTION

1.1 General

Electricity is one of the most important inventions of science to humanity. It has become a part of modern life, and life without it is unthinkable. It has a significant position in the industry and its importance in individual use in daily life. It has an important function, especially in terms of the electricity consumption of electric motors [1]. Electric machines cover most of the energy consumed are the main building blocks for industries. The scope of uses goes from modern assembling and preparing to business and homegrown climate used to convert electrical power to mechanical energy with very high efficiency [2-3].

Electric motors such as blenders, fans, pumps, blowers, transports, plants, cranes, vehicles, machine apparatuses in industrial applications consume nearly 25% of the world's energy [4]. This energy consumption is increasing day by day. Variable frequency drive (VFD) systems [5-7] can be more beneficial because of cooling systems and damper control of liquid flow. Also, VFDs for all applications have a pie in the global market [8]. A VFD cost and size has reduced with power electronics technology developed performance in drive topologies, control hardware, and software in the past forty years. It controls the input voltage according to the duty cycle and frequency of a motor. In this way, the speed-dependent on the frequency of the engine is also influenced by the VFD [9-10].

The switching frequency of a VFD refers to the rate at which the DC bus voltage is turned on and off during the pulse width modulation (PWM) process. Insulated gate bipolar transistor (IGBTs) are used to turn on and off the DC voltage. The PWM process makes use of the IGBT switching to generate the variable voltage and frequency output from the VFD for controlling AC induction, permanent magnet

synchronous, or direct current motors. As the switching frequency of the PWM process increases, the harmonic content of the current waveform generated by the PWM process decreases. By reducing current ripple, the 'cleaner' waveform results in increased efficiency and consequently lower motor losses. This advantage of a higher switching frequency becomes more pronounced as the motor's output frequency increases. Reduced current distortion results in less rotor heating and increased motor efficiency. Reduced rotor heating is a significant issue when motors employ bearing technology that necessitates extremely small clearances (air foil or magnetic). Excessive rotor heating can cause the rotor to expand or elongate, potentially colliding with the bearing surface.

The increased switching frequency reduces the audible noise produced by the motor. The motor's audible noise is caused by stator laminations vibrating at the carrier frequency rate. As the carrier frequency is increased, the pitch of the noise generated by the stator laminations increases, pushing the levels beyond the range of human hearing. Motor noise levels may be an issue depending on the application (elevator motors, theater equipment, etc.). In these instances, a variable frequency drive (VFD) with a higher carrier frequency may be an option.

Motor heating is reduced as the switching frequency increases due to the presence of more harmonics in the current waveform. Simultaneously, the amount of heat generated internally by the VFD as a result of the IGBT switching is increased. Each IGBT switching action results in a relatively constant amount of heat loss. Thus, as switching frequency increases, the VFD's overall heat loss decreases. The VFD's heatsink must be designed in such a way that it can operate at the maximum rated ambient temperature. Due to the increased heat loss caused by the higher switching frequency, as well as the fact that the heatsink may need to be larger (if air cooled), this results in a larger physical size VFD and thus higher initial component costs.

The ratings for VFDs are based on the most extreme ambient conditions. The VFD is typically mounted inside an electrical enclosure. As a result, the temperature inside the enclosure is equal to the maximum ambient temperature surrounding the

VFD. Increased heat loss from the VFD as a result of the higher switching frequency results in increased heat loss into the enclosure. The increased heat load on the enclosure may necessitate the addition of cooling (fans, air conditioner) to the enclosure itself, depending on the application requirements.

The switching frequency of a VFD used in a particular application is highly dependent on the application requirements and system components. When determining the optimal VFD switching frequency for an application, the switching frequency of a VFD can be changed, typically between 1 and 20kHz. Choosing an appropriate carrier frequency involves balancing the positive and negative effects generated by this factor. The positive effects are as follows when the VFD switching frequency is increased: cleaner output waveform, lower audible noises on the motor, the higher efficiency of the motor and allow reaching higher ratio speed of the motor shaft. On the other hand, the negative effects are follows in the same condition: increases heating of the drive, reduces the lifetime of the VFD, increases voltage spikes that may influence insulation of the windings in the motor and increasing EMI noises in the system. In this thesis, low switching frequency range have been studied to reduce negative effects and frequencies below 2 kHz have been tested.

A simple circuit of the VFD is represented in Figure 1.1. The VFD contains three main parts: rectifier circuit converted to DC power to AC power, inverter circuit, and motor part. The DC unit called DC-Link filtered and stored energy in the high power capacitors, and the inverter circuit converted DC power to AC power and controlled the motor.

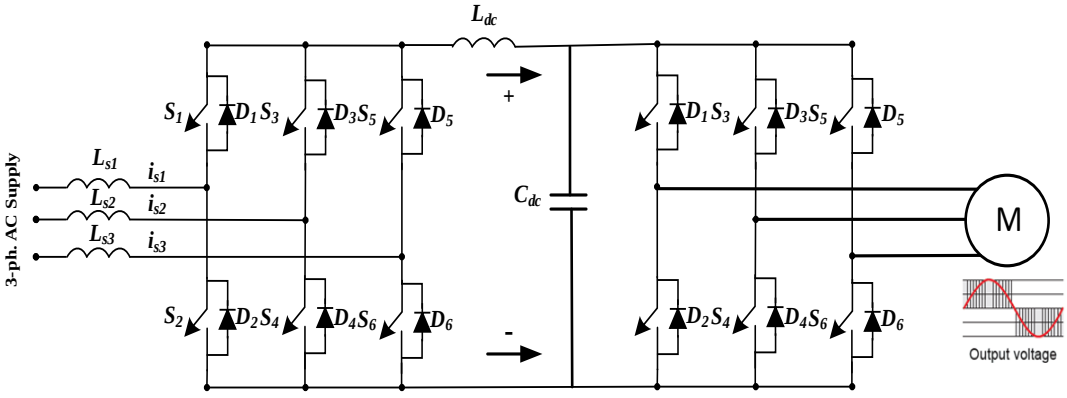


Figure 1.1: A Simple Diagram of Variable Frequency Drive

The use of new technology switches such as insulated-gate bipolar transistors (IGBTs) Generation 7 in motor drive circuits provides 25% small driver circuit as a result of using more minor chips, ability to work in harsh environmental conditions, low switching power loss, maximum output voltage and long life [11]. Examples of other semiconductor families can be silicon carbide (SiC) metal-oxide-semiconductor field-effect transistors (MOSFETs) shown in Figure 1.2 and diodes with low switching losses at high junction temperature and high power density in low switching frequencies [12].

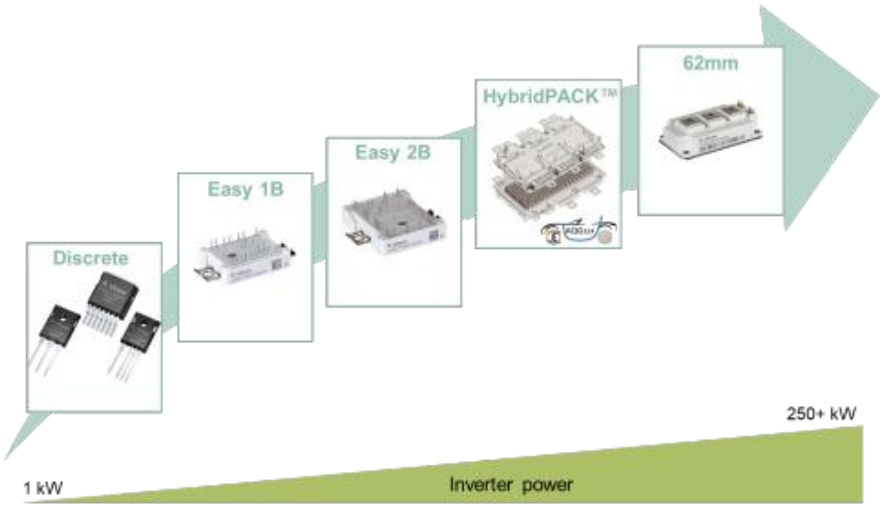


Figure 1.2: Product Line Up According to Output Power Ratings

The control of VFD has become more crucial and various with the improving microcontroller technologies. The increasing speed and power of microprocessors prepare the infrastructure for real-time implementation of complex and high computational power applications. Field-programmable gate arrays (FPGAs) and graphic processing units (GPUs) can be given as examples of microprocessors.

FPGAs [12] are gadgets involving programmable rationale blocks which can be arranged to perform distinctive rationale capacities. The rationale blocks are associated with electronic wiring that makes for the inside directing arrangement of the chip – interconnects — that can be turned here and there. An FPGA can be reconfigured by programming the rationale impedes and controlling the inside directing. This way, FPGAs can be reconstructed even get-togethers, making them

ideal for frameworks and gadgets that need continuous updates like models, organizing items, and other electronic frameworks [13-14].

GPUs were initially intended for illustrations just, yet have upgraded to become viable for use with various applications in all cases. During the 1980s, they were used to offload illustrations from the CPU, which used to be amazingly basic at that time. As we advanced, drawings started to be further developed, with the presentation of high goal pictures, 2D and 3D imaging, and video preparing, which implied that GPUs also needed to become progress [15-16]. Each image is made out of thousands of pixels handled by many indistinguishable centers that are explicitly intended to execute various capacities identically. Due to their very effective equal working, GPUs are currently being utilized in a wide range of fields and applications, including a portion of the world's quickest ever supercomputers for the execution of different numerical capacities simultaneously [17]. A GPU is a chip that performs fast numerical computations, fundamentally with the end goal of the video. It comprises countless sluggish and quick processors that are working in equal. GPUs can process vector math, grid math, pixel changes, and delivering occupations around 10-500x quicker than the same CPU execution [18-19].

1.2 Inverters

Inverters, DC to AC converters are widely used for VFD AC drives [20-22], photo-voltaic (PV) applications [23-24], uninterruptible power supplies [25-27], high voltage DC applications [28], and many other industrial applications [29-32]. The inverter design determines the inverter input voltage, output voltage, operating frequency, and overall power. Without a DC input source, the inverter cannot produce an AC output voltage. DC sources can be batteries, solar or fuel cells, and rectifiers. According to the duty of the inverter, this DC source voltage is converted to desired AC output voltage magnitude and frequency. An inverter can produce sine wave, modified sine wave, pulsed sine wave, pulsed width modulated (PWM) wave, and square wave. Although the inverter output voltage is pure sine wave in ideal, it

is impossible because of the switching of semiconductors. The switching of semiconductors causes specific harmonics on the inverter output voltage in practice.

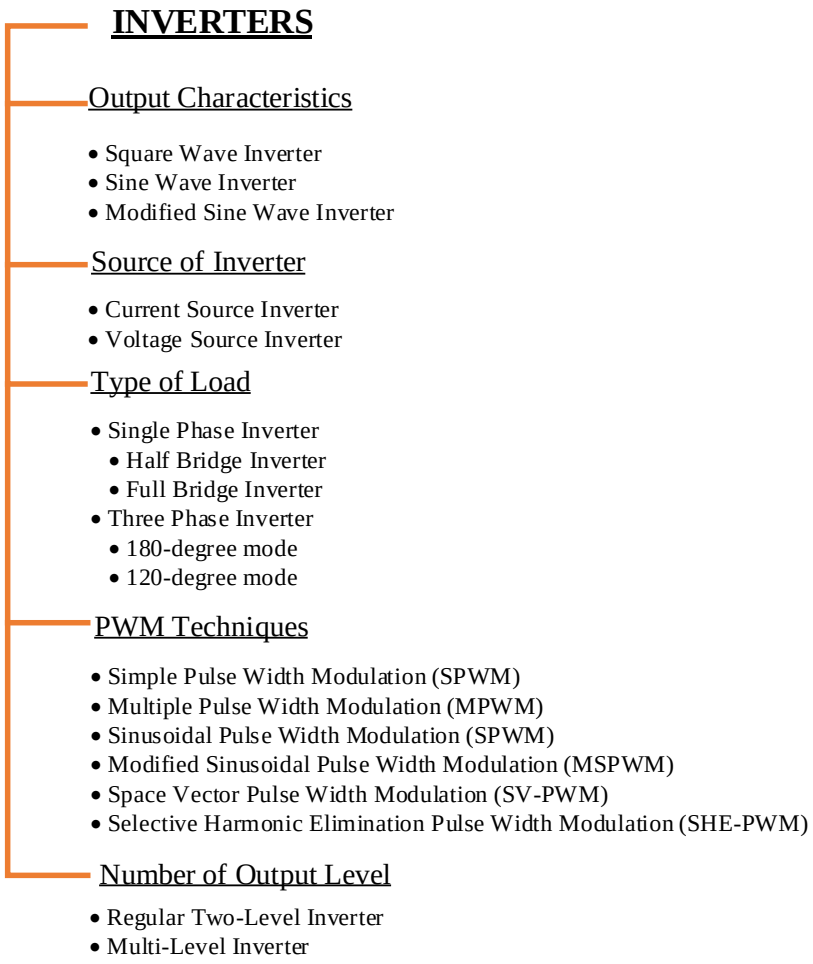


Figure 1.3: Classification of Inverters [33]

Square wave and quasi-square wave voltages can be taken in low and medium voltage applications, but lowly-distorted sine wave is required in high power applications. Advanced switching techniques and high-speed power semiconductor devices reduce the harmonic content of the inverter output voltage [34-35]. Inverters can be classified according to source, types of output, load types, etc., and this classification of the inverters shown in Figure 1.3 is explained. This thesis examined the sine-wave voltage source three-phase two-level inverter and compared it with

three of the most efficient PWM techniques, SPWM, SVM-PWM [36], and SHE-PWM.

Sinusoidal pulse with modulation (SPWM) implementation [37] presented in Figure 1.4 is easy with an analog circuit including a comparator. Specifically designed courses produce sine and triangle waves to feed the comparator circuit, making the desired SPWM output voltage signal. But the reliability and control precision of the scheme given in Figure 1.4 do not perform perfectly because of the complex circuit architecture, so analog devices parameters are unstable. Nowadays, the software implementation of SPWM allows high precision control with the enhancement of microcontroller technology.

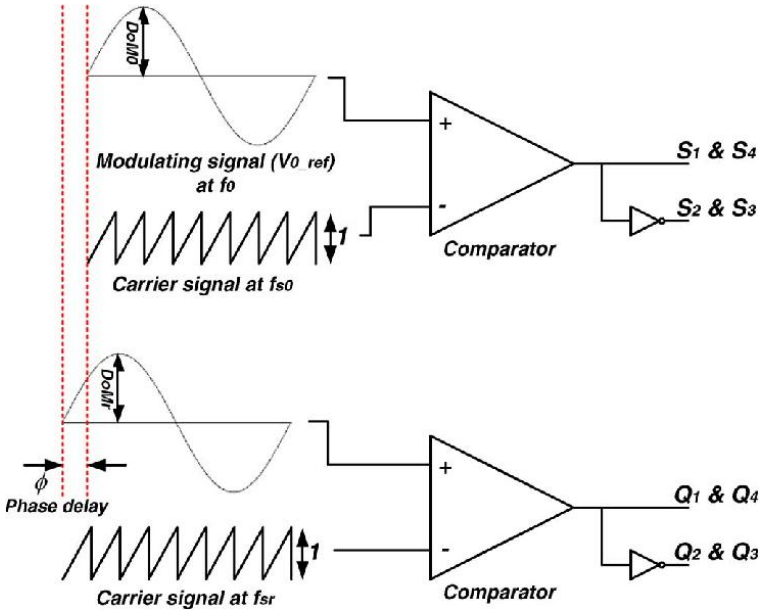


Figure 1.4: Analog Scheme for SPWM Implementation

The strategy using the regular convergence points of a sine wave with triangle wave to acknowledge PWM is classified as "Natural Sampled Method." It's ready to exhibit the genuine minutes the beat is begun and finished, and the SPWM waveform is a lot nearer to a sine wave. This strategy isn't embraced in most control applications because of the arbitrary convergence points of sine and triangle waves, bringing about convoluted estimation and troublesome ongoing execution. To conquer these detriments, another new technique called "Regular Sampled Method" was advanced.

It is broadly utilized in designing applications these days. It depends on the rule that a specific second is chosen in each pattern of the triangle transporter wave to track down the comparing worth of the sine wave voltage, which is acquainted with the test on the triangle wave. The example result decides the ON/OFF snapshots of the force gadgets, disregarding whether the sine wave and triangle wave meets at this time or not. A more viable strategy named "Normal Symmetric Regular Sampled Method" (represented in Figure 1.4) is applied in most control cases.

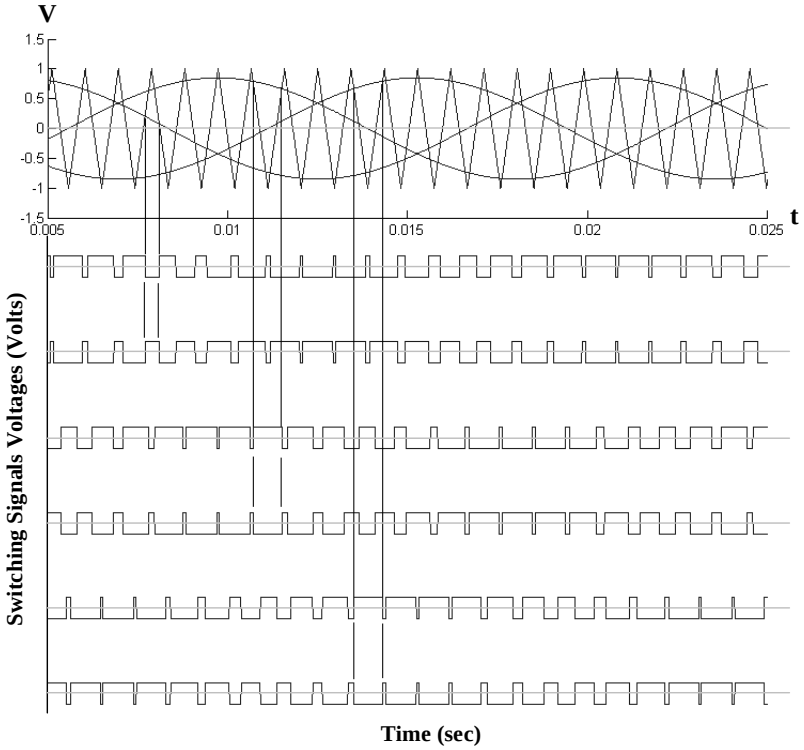


Figure 1.5: SPWM method gate signals

In Figure 1.5, the tested second is given on the box point of the triangle wave, then, at that point, focused by the comparing worth of sine wave voltage. A level line is attracted to cross the triangle wave on the two sides, so the primary and following edges of the PWM waveform are chosen that. The leading edge is a little more extensive, which repays the next thin edge. Accordingly, as a normal thought, the

viability of this technique is practically identical to that of the regularly examined strategy.

Because of the 3-stage coordinated age viability and moving toward the best-adjusted track with a steady abundancy of the pivoting field framed by the hole transition, the space vector pulse width modulation (SVPWM) [38-39] waveform is acknowledged by consolidating diverse exchanging methods of the inverter. In a 3-stage inverter, if "1" is characterized as the positive portion of the DC bus voltage and "0" as the negative half (both are alluded to as the neutral point). There are eight switch states for the six force switches represented in Figure 1.6. Consequently, eight voltage vectors (dynamic vectors $\overrightarrow{U_1}$ ~ $\overrightarrow{U_6}$ and zero vectors $\overrightarrow{U_0}$, $\overrightarrow{U_7}$) can be correspondingly characterized to shape the vector space, separated into six areas Figure 1.7.

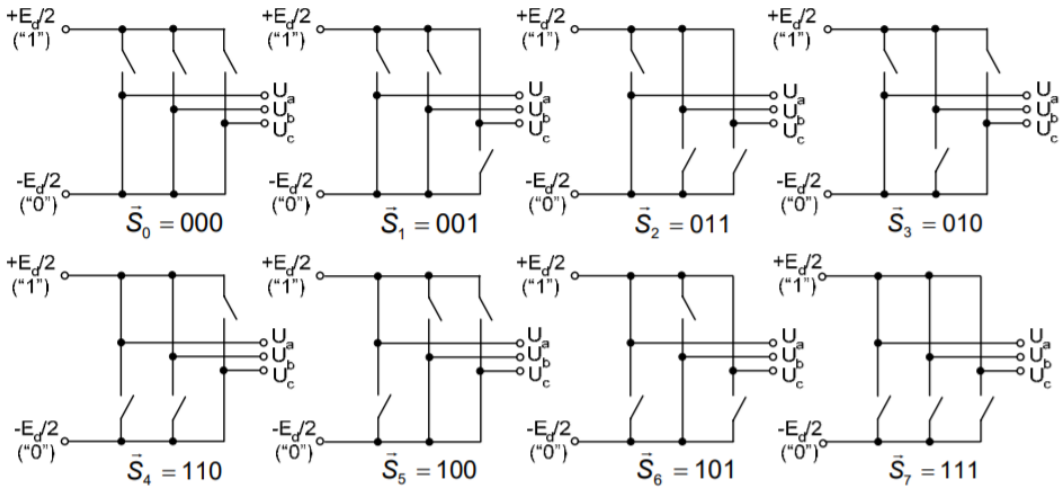


Figure 1.6: Eight Switch States

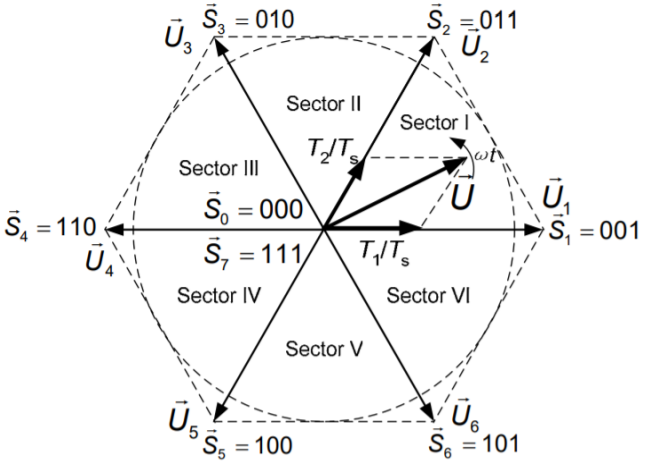


Figure 1.7: Voltage Vector Space

The voltage vector $\vec{U}$ is, for the most part, deteriorated into two closest contiguous voltage vectors with zero vectors $\overrightarrow{U_0}$ as well as $\overrightarrow{U_7}$ as supplement. Consequently, the vectors in the six areas voltage and current vectors are moderately determined (expect that the zero vector activity time is divided by $\overrightarrow{U_0}$ and $\overrightarrow{U_7}$) as the conditions Equation 1.1:

$$\overrightarrow{U_I} = \frac{T_1}{T_s}\overrightarrow{U_1} + \frac{T_2}{T_s}\overrightarrow{U_2} + \frac{T_s - T_1 - T_2}{2T_s}(\overrightarrow{U_0} + \overrightarrow{U_7})$$

$$\overrightarrow{U_{II}} = \frac{T_2}{T_s}\overrightarrow{U_2} + \frac{T_3}{T_s}\overrightarrow{U_3} + \frac{T_s - T_2 - T_3}{2T_s}(\overrightarrow{U_0} + \overrightarrow{U_7})$$

$$\overrightarrow{U_{III}} = \frac{T_3}{T_s}\overrightarrow{U_3} + \frac{T_4}{T_s}\overrightarrow{U_4} + \frac{T_s - T_3 - T_4}{2T_s}(\overrightarrow{U_0} + \overrightarrow{U_7}) \quad (1.1)$$

$$\overrightarrow{U_{IV}} = \frac{T_4}{T_s}\overrightarrow{U_4} + \frac{T_5}{T_s}\overrightarrow{U_5} + \frac{T_s - T_4 - T_5}{2T_s}(\overrightarrow{U_0} + \overrightarrow{U_7})$$

$$\overrightarrow{U_V} = \frac{T_5}{T_s}\overrightarrow{U_5} + \frac{T_6}{T_s}\overrightarrow{U_6} + \frac{T_s - T_5 - T_6}{2T_s}(\overrightarrow{U_0} + \overrightarrow{U_7})$$

$$\overrightarrow{U_{VI}} = \frac{T_6}{T_s}\overrightarrow{U_6} + \frac{T_1}{T_s}\overrightarrow{U_1} + \frac{T_s - T_6 - T_1}{2T_s}(\overrightarrow{U_0} + \overrightarrow{U_7})$$

1.3 Selective Harmonic Elimination

Overall system efficiency is critical in high power motor drives. Thus, switching and harmonic losses must be considered in order to minimize operational costs and thermal strains (e.g. longer life time and higher operational system efficiency). As a result of these constraints, IGBT switching rates are limited to 500-3000 Hz. Modulation index becomes extremely low at lower switching frequencies, particularly in the high speed operation zone. This decreases the precision of PWM patterns and may result in the loss of symmetrical full-wave, half-wave, and quarter-wave waveforms. As a result, the output waveforms contain low order (5th, 7th, 11th, 13th, 17th, 19th etc.) frequency harmonics and these harmonics result in large torque ripple. In practice, modest passive filters cannot be used to filter low frequency harmonic components. In this case, the filter increases the size, weight, and expense of the product. Additionally, numerous studies demonstrate that multilayer converters work well in reducing harmonics in high power applications. However, this raises the cost and reduces the converter's reliability due to the increased number of semiconductors, gate-drivers, capacitors, and isolated power supply, among other components. On the other hand, various PWM approaches may aid in the management of low order harmonics up to a point. In practice, three basic programmed PWM methods are utilized to eliminate low order frequency harmonics, including Selective Harmonic Elimination PWM, Selective Harmonic Mitigation PWM, and Synchronous Optimal PWM. It will be extremely handy to apply these efficient modulation method to create output waveforms with an acceptable harmonic content in high power applications. These modulation approaches are based on the cancellation of low order current harmonics or the minimizing of the RMS value of the motor's current harmonics by a careful selection of commutation angles using appropriate objective functions.

The SHE-PWM method addresses a progression of trigonometric conditions obtained from the Fourier extension of the inverter output voltage [40]. The arrangement is the exchanging points that will put explicit low order frequency

harmonics equivalent to nothing while controls the fundamental frequency component [41]. Figure 1.8 shows a regular symmetric and ventured voltage waveform integrated by a 2S +1 level inverter, where S is the quantity of exchanging points and the quantity of DC sources. For equivalent dc sources (V_{dc}), the Fourier series extension of the output voltage waveform, displayed in Figure 1.8, is calculated from Equation 1.2:

$$E_{out}(\omega t) = \sum_{n=1,3,5,\dots}^{\infty} \frac{4V_{dc}}{n\pi} \begin{bmatrix} \cos(n\theta_1) + \\ \cos(n\theta_2) + \\ \vdots \\ \cos(n\theta_S) \end{bmatrix} \times \sin(n\omega t) \qquad (1.2)$$

Ideally, with a given desired fundamental voltage V_1, it is possible to determine the switching angles $\theta_1, \theta_2 \dots, S$, and specific harmonics are equal to zero. It has been proved that S equations are needed to control the fundamental output voltage and eliminate $S-1$ harmonics. For example, a 9-level inverter can handle the fundamental component and eliminate the amplitudes of three harmonics. The switching angles can be found by solving the following equations:

$$cos(\theta_1) + \cos(\theta_2) + \cdots + \cos(\theta_S) = m$$

$$cos(n\theta_1) + \cos(n\theta_2) + \cdots + \cos(n\theta_S) = 0,\ n = 5,7,11 \qquad (1.3)$$

where $m = \frac{\pi V_1}{4V_{dc}}$.

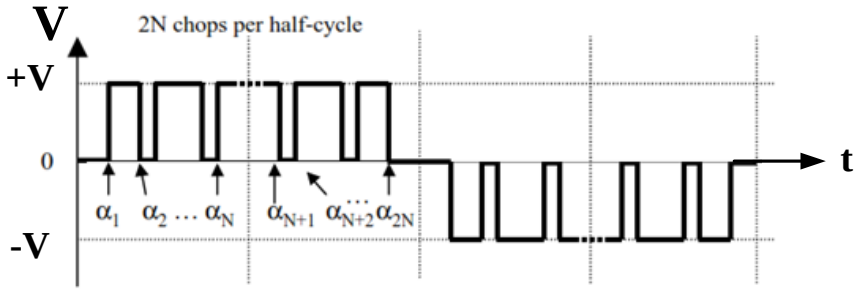


Figure 1.8: Two-level VSI Voltage Waveform Applied to SHE-PWM

Nowadays, another popular power converter is the multilevel inverter which converts from several DC input voltages to needed AC output voltage. With a sufficient number of DC sources, it is possible to synthesize a nearly sinusoidal voltage waveform. Compared to hard-switched two-level [42] pulse width modulation inverters, multi-level inverters [43] have some features, such as high efficiency, low electromagnetic interference, which can operate at the high voltage at lower dv / dt per switch and offers the benefits of [44-46]. To combine multiple levels of output AC voltage using different levels of DC input, power on and off the semiconductor device to get the essential elements needed for the lowest harmonic distortion.

A commonly available switching technique is the SHE method at the fundamental frequency, which solves the transcendental equations that characterize the harmonics and calculates the switching angle [47], [51]. Solving the SHE equation is difficult because the SHE equation is very non-linear and may not be simple or have created multiple or all solutions for a particular modulation index value. The big challenge is how to obtain a set of all solutions that can exist using a simple, computationally less complex method.

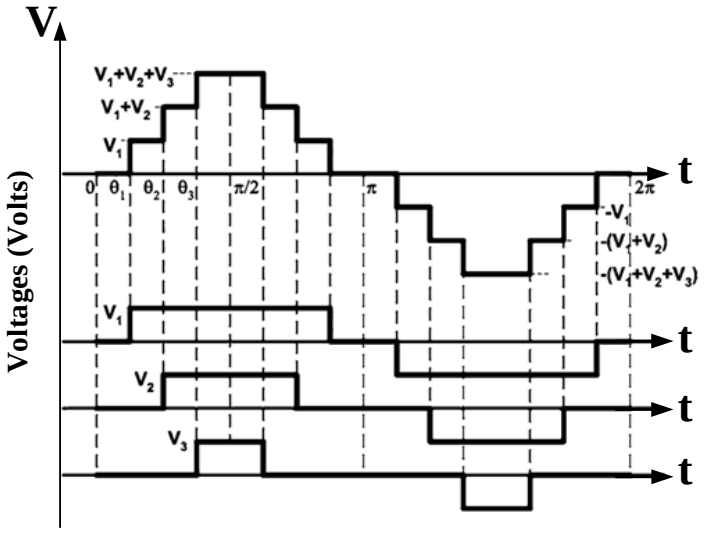


Figure 1.9: Seven-level VSI Voltage Waveform Applied to SHE-PWM

The two-level and multilevel inverter topologies are used in a wide variety of industrial applications. An inverter can create a controlled voltage for a VFD and act as a motor drive in the steady-state. SHEM can be lower the harmonic content of the inverter output, hence increasing the system's efficiency [52]. The paper [53] describes a two-level three-leg voltage-source converter (VSC) design and implementation in light-rail traction inverter applications. SHEM eliminates the output voltage's 5th, 7th, 11th, 13th, 17th, and 19th harmonics. Another motor driving application utilizing SHEM is discussed in [54]. The SHEM approach is used to identify five switching angles that eliminate the fifth, seventh, eleventh, and thirteenth harmonics while maintaining a low harmonic distortion of the line-to-neutral output voltage [55]. As a result, no additional series filter reactor is required at the output. Improved current waveforms can be achieved in practical systems by increasing the number of harmonics to be eliminated (usually, up to 29th harmonics for three-phase systems). Additionally, when SHE is used in medium and high voltage applications, the converter switching devices operate at a low frequency, resulting in minimal switching losses.

In both the two-level and multi-level inverter instances, a series of nonlinear equations must be solved to determine the switching angles of SHEM [56-60]. These equations are primarily composed of sinusoidal terms, and the set's total number of equations equals the total number of switching angles. While increasing the switching angles eliminates more harmonics, the complexity of the problem increases as the switching angles increase, and finding the solution set by conventional approaches becomes extremely difficult.

In the SHE-PWM technique, advanced angles are chosen by a collection of non-direct circumstances that exclude undesired low-order harmonics from the inverter's output voltage. Several SHE-PWM methods were developed in the literature to identify increased progress focused on real-time, limiting or eliminating undesired low-order harmonics. The SHE-PWM approach is divided into three critical classes [61]:

1. Numeric Methods (NMs)
2. Arithmetic Methods (AMs)
3. Bio-Intelligent Algorithms (BIAs).

NMs are fast iterative methodologies that calculate optimal solutions in a limited number of iterations. However, these methods rely on initial guesses and become stuck at the local minimum solution without reasonable guesses [62]. This method is frequently used thanks to its capability to determine exact and precise solutions [63-70]. Predictive [71] and Bayesian [63-67] systems have been suggested for estimating the initial values of NR. However, this hybrid method becomes more computationally complex when estimates for high-level inverters, particularly for the type-b output voltage waveform. Walsh function [72-73], homotopy algorithm [74-75], sequential quadratic programming [76], and gradient optimization [77] are additional improved NMs described in the literature. However, these methods do not address the initial guess problem.

AMs convert nonlinear transcendental equations to polynomial equations to obtain the optimal firing angles. The primary benefit of this technique is that no initial guess is required. The resultant theory [78-80], the Wu method [81], the Groebner bases theory [82], the symmetric polynomial theory [83], and the power sum [84] methods are all used in the literature to calculate SHE in inverters. These methods, however, are computationally intensive and are appropriate to only low-level inverters. As a result, this method is not suitable for use with inverters in real-world applications.

BIAs are artificial intelligence techniques that are influenced by natural phenomena such as species relocation, natural selection, ant colonization, human culture, pollinator colonies, bird swarms, and fish schools [85-89]. BIAs are iterative population-based techniques for which initial guesses have a negligible effect on discovering optimal solutions. BIAs are straightforward to comprehend and straightforward to program on personal computers utilizing MATLAB or other computational software. BIAs employ an optimization problem that contains transcendental formulas for the fundamental and low-order harmonics. This

technique optimized the firing angles by minimizing the objective function. The achievement of BIAs is highly dependent on how an objective function is defined. Each researcher primarily creates their optimal solution to accomplish their objectives. The following section discusses the most popular BIAs in detail, including the Bat Optimization Algorithm (BOA), BA, PSO, Firefly Algorithm (FA), Cuckoo Search Algorithm (CSA), GA, Imperialist Competitive Algorithm (ICA), Artificial Neural Network (ANN), and DE. Several other BIAs have been described in the literature, including Ant Colony Optimization [90], Clonal Search Algorithm [91], Bacterial Foraging Algorithm [92-93], and Shuffled Frog Leaping Algorithm [94]. These algorithms, however, are not frequently used to solve the HE problem. Additionally, they have limited capabilities; thus, they are omitted from the detailed explanation.

Apart from the categories mentioned previously, several novel approaches to HE, such as modulation-based harmonic elimination [95] and four-equation-based harmonic elimination [96-98], have been proposed in the literature.

The algorithms for solving SHEM equations are summarized in Figure 1.4.

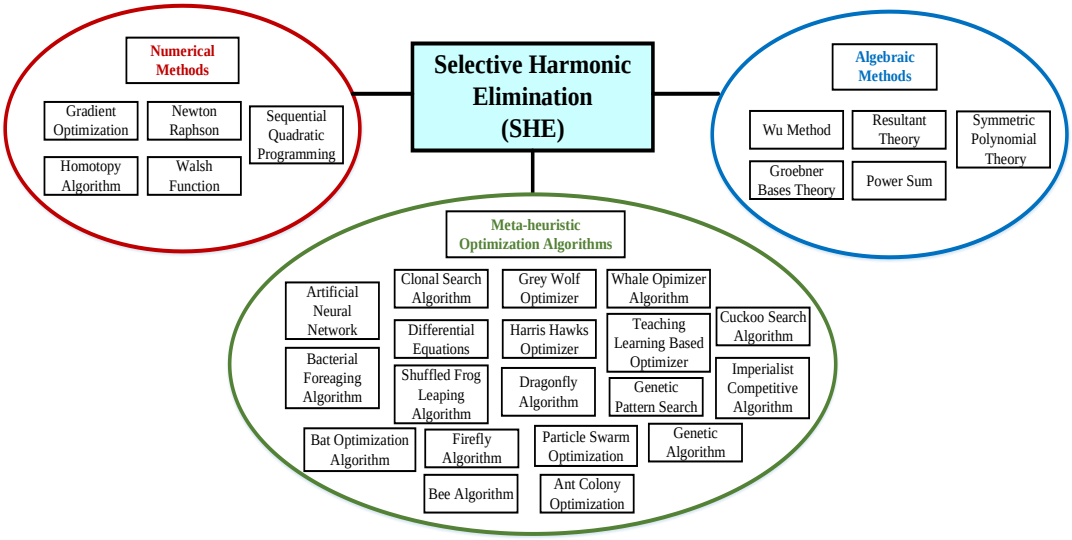


Figure 1.10: Classification of SHE Method Algorithms

Compared to a 7-level inverter, a 5-level inverter with only an optimum dc source in [99] is more successful over all modulation indexes when simulated results, cost, and implementation are considered. Additionally, the results demonstrate that GPS outperforms genetic algorithms (GA) to obtain precise global minima. To enable a more accurate evaluation of the benefits of optimizing additional dc sources and GPS performance compared to GA, a seven-level inverter with two optimal dc sources is used to eliminate the fifth, seventh, eleventh, and thirteenth harmonics [100]. To facilitate comparing those methods and the effect of optimal dc sources, 5-, 9-, and 15-level inverters are selected and optimized using MATLAB software. The simulation results demonstrate that teaching-learning-based optimizer (TLBO) is superior to the other algorithms in terms of accuracy and possibility of convergence [101].

Selective harmonic elimination is reformulated in [102] as an optimization problem in conjunction with output voltage regulation, and the solution is sought using a novel concept. It is demonstrated that a recently launched parameter in the algorithm—the quantum of ants' step movement—has a significant effect on the algorithm's speed of convergence.

SHE is a highly efficient technique for obtaining the desired fundamental aspect while avoiding selection harmonics. The SHE problem's nonlinear transcendental equations are solved using a newly invented evolutionary optimization method called the BAT algorithm [103]. The article [104] discusses a fast convergent, accurate coded genetic algorithm (HRCGA) that significantly reduces the computational burden. They derive an objective function that describes the effectiveness of removing selected orders of harmonics while maintaining control of the fundamental. We report on a comparison of various operating points.

The selective harmonic elimination (SHE) technique, which is based on a firefly-assisted genetic algorithm (FAGA), is being developed to efficiently control and decrease the harmonics from the planned system [105]. Using the FAGA algorithm, the desired low-order harmonics from the output voltage of the photovoltaic System

are removed from the output voltage. The currently available firefly algorithm (FA) and genetic algorithm (GA) are also built to determine whether FAGA is superior to them. The incremental conductivity (IC) algorithm is also used to ensure that the PV system produces the maximum amount of power.

The research [106] proposes using a bacterial foraging algorithm (BFA) to select the switching angle in a PWM inverter. The problem of voltage harmonic elimination in conjunction with output voltage regulation is formulated as an optimization task, and the proposed method is used to find a solution. This article [107] discusses novel applications of the Firefly and Fireworks algorithms to the problem of selective harmonic reduction in inverter output waveforms. The success of an algorithm is determined by many factors, including the rate of convergence, the number of iterations, and the accuracy.

The Grey Wolf Optimizer (GWO) method is used in this article [108] to discover the optimal switching angles for a cascaded multilevel inverter capable of removing some high-order harmonics while preserving the requisite fundamental voltage. In [109] paper, GWO is applied to the two-level three-phase inverter for switching optimum switching angles.

The Whale Optimizer Algorithm (WOA) is used in this article [110] to determine the optimal switching angles for a three-phase Voltage Source Inverter (VSI) to eliminate some high order harmonics while maintaining the needed voltage. Additionally, the Particle Swarm Optimization (PSO) algorithm is used, and the optimal switching angles are determined off-line to eliminate the fifth, seventh, eleventh, thirteenth, thirteenth, seventeenth, and nineteenth harmonics. The output voltages produced by WOA and PSO are compared in terms of total harmonic distortion (THD). By comparing these results, we demonstrate that WOA has more accurate and faster outcomes in reducing THD than the PSO method in SHE applications.

1.4 Contributions

In this thesis, real-time and offline SHE switching angles calculations using PSO, GWO, WOA, and HHO are realized and tested on the traction inverter motor drive. The real-time SHE changing angle calculations are applied on Titan V and Jetson TX1 GPUs. But, these cards could not be implemented in the motor drive system because of funding and infrastructural problems. Using lookup tables, the offline switching angles are implemented on the light-rail system on the Texas Instruments TMS 28377 DSP board.

From this thesis, contributions below can be sequenced:

- SHE-PWM primary objective is to reduce filtering needs in order to reduce the size, weight, and cost of filtering devices while still meeting harmonic criteria. In this thesis, various tests were carried out with variable harmonic numbers N up to 17 (50th harmonic). As N value increased, THD and TDD values decreased. In addition, the usage of SHE-PWM decreases heating of the drive, increases the lifetime of the VFD, decreases voltage spikes that may influence insulation of the windings in the motor and decreasing EMI noises in the system. On the other hand, when the N value was increased to 17, the problem became more complex, the solution time and the switching loss increased. Therefore, a new microprocessor may be needed.
- The HHO algorithm is first used to calculate the switching angles of a two-level three-phase motor drive system using SHEM and eliminated low-order harmonics in this thesis.
- The HHO algorithm is evaluated in comparison to three well-known and influential algorithms: PSO, GWO, and WOA. HHO outperforms other algorithms in terms of convergence rate and accuracy of SHE applications.
- In Matlab/Simulink, the calculated SHEM angles for these four methods are applied to a two-level three-phase motor drive system. Additionally, the system is tested with various DC-Link voltages and fundamental frequencies

rather than eliminating low-order harmonics. In order to obtain the desired output voltage level, the modulation index changes as the dc link voltage level changes. The modulation index is defined between 0.1 and 0.95, and the switching angles cannot be calculated correctly in modulation index values outside this range.

- Furthermore, SPWM, SVPWM, and SHEM are compared at low-switching frequencies in Matlab/Simulink environment. The performance of these three methods varies depending on the number of harmonics N eliminated at low frequencies. For N=9, SHE-PWM becomes comparable to other methods in terms of TDD. At higher N values, SHE-PWM performs better than SVPWM and SPWM in terms of harmonic elimination.
- HHO switching angles are implemented to a light-rail transportation system in the laboratory setup firstly in the literature and eliminated desired 5th, 7th, 11th, 13th, 17th and 19th harmonics.
- Titan V and Jetson TX1 GPUs are used to be a platform run CUDA PSO, GWO, WOA, and HHO codes. All the calculations of GPUs are realized approximately 200x faster than computer CPU.

1.5 Outline of the Thesis

SHEM was used in the lookup table in this thesis to determine optimal switching angles via harris hawks optimization compared by particle swarm optimization, grey wolf optimization, and whale optimizer algorithm, thereby removing 5th, 7th, 11th, 13th, 17th, and 19th harmonics in the output voltage of a variable frequency motor drive given in Figure 1.9 inverter with a changing DC link input voltage. The HHO algorithm was implemented in the TMS28377 DSP board, which served as the primary and single controller, providing gating signals for the inverter's SiC MOSFETs. Furthermore, SHEM was used to find optimum switching angles in Titan V and Jetson TX1 GPUs and compared four metaheuristic algorithms on these units.

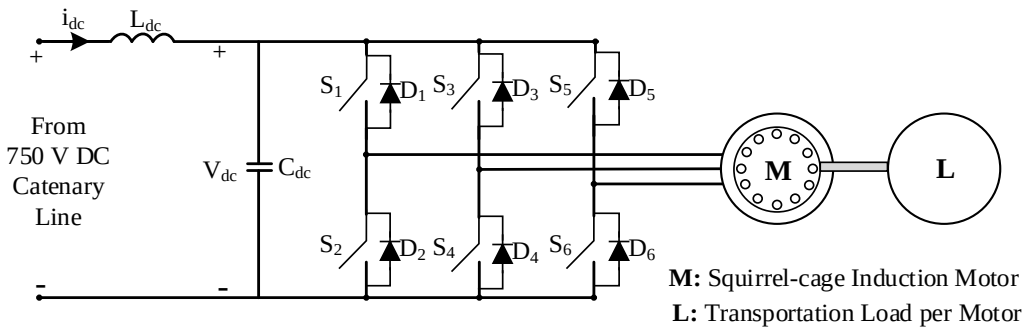


Figure 1.11: Variable Frequency Motor Drive

The following is the outline of this thesis:

Chapter 2 defines the problem for SHEM's offline application. The variable DC link voltage inverters are researched in two specific applications: solar and traction. The modulation index values for inverters with varying DC connection voltage are found to vary widely. The difficulties of computing SHEM switching angles offline and then storing them in the microcontroller as lookup tables are demonstrated by examining the discretization of modulation index values, the storage requirements for lookup tables, and the infeasible region of solution space.

Chapter 3 discusses metaheuristic algorithms, including PSO, GWO, WOA, and HHO are explained in detail. The philosophy underlying metaheuristic algorithms are discussed and their common characteristics and advantages over classic iterative methods. These algorithms are then introduced, along with the process flow and equations employed in practice. SHEM patterns are discussed in detail, and the four algorithms are used to solve the specified SHEM problem's cost function.

Chapter 4 details the light-rail transportation definition and simulations of PSO, GWO, WOA, and HHO algorithms' switching angles calculations. Implementation of these algorithms in the MATLAB Simulink platform is explained in detail and presented tables, including switching angles eliminated 5th, 7th, 11th, 13th, 17th, and 19th harmonics. These angles are tested on the simulated system with different modulation index values and different frequencies. HHO algorithm shows better properties than other metaheuristic algorithms.

Chapter 5 explains used GPU systems, including Titan V and Jetson TX1 properties. This chapter evaluates GPUs, GPU computing trends, CUDA programming and compares these cards' performance results, including PSO, GWO, WOA, and HHO algorithms real-time calculation speed of SHE-PWM switching angles. According to experimental results, online calculations of SHEM can be realized on GPU platforms using metaheuristic algorithms.

Chapter 6 discusses the SPWM, SVPWM, and SHE-PWM methods of a two-level three-phase traction motor drive system. In low frequencies up to 1500 Hz, SHE-PWM can eliminate low-order harmonics. Therefore, overall SHE-PWM THD values are lower than SPWM and SVPWM techniques.

Chapter 7 discusses traction inverter systems using lookup table SHEM switching angle calculations. Finally, experimental findings for the light-rail three-phase two-level inverter are presented with appropriate figures, tables, and statistics.

Chapter 8 summarizes the work completed for the thesis and makes some predictions and recommendations for future work.

Appendix A contains the CUDA PSO code. Similarly, Appendix B, C, and D include GWO, WOA, and HHO CUDA codes.

CHAPTER 2

PROBLEM DEFINITION

SHE-PWM applications can be categorized into two parts to using online (real-time) and offline techniques. In this chapter, the application of offline SHE-PWM will be explained. The calculation of offline SHE-PWM switching angles has some disadvantages: variable DC-link voltage of three-phase inverters, modulation index discretization, the storage of lookup table, and impracticable region of SHE implementation space.

2.1 Variable DC-Link Voltage

As previously stated in the first chapter, inverters generate an alternating current voltage from direct current voltage, with the frequency and magnitude of the output voltage being controlled by the inverter. The modulation index is the most critical parameter in an inverter, and it varies based on the topology characteristic and control mechanism of the inverter. The modulation index regulates the output voltage of the inverter, which regulates reactive and active power according to the demands of the load. This array's top and bottom semiconductors are switched to link either + Vdc / 2 or -Vdc / 2 to the output, and the reference is taken to be the midpoint of the input capacitor in Figure 2.1. Thus, the DC input voltage is transformed into an alternating current output voltage alternating between the positive and negative phases. Figures 2.2 and 2.3 [111] show how single-phase full-bridge inverters and three-phase full-bridge inverters can be formed from this architecture by including semiconductor legs in the circuit.

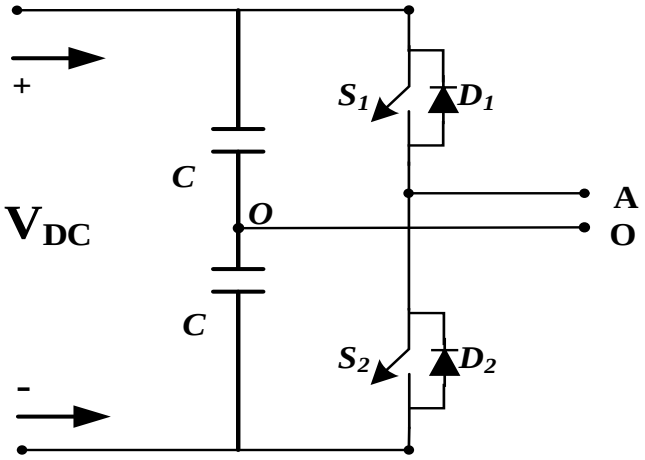


Figure 2 1: Single-Phase Half-Bridge Inverter

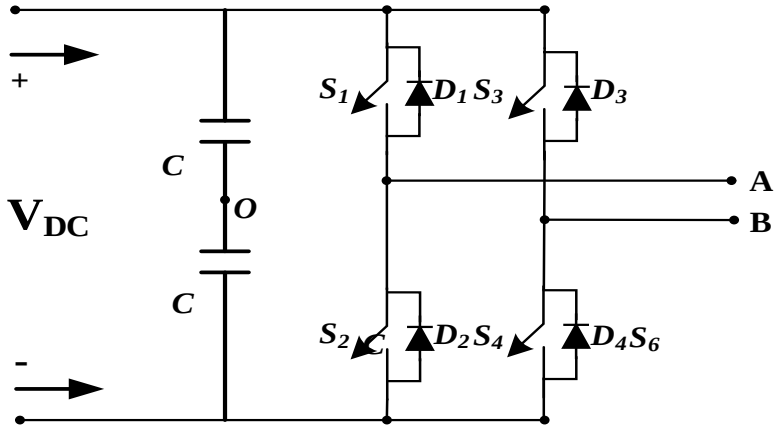


Figure 2.2: Single-Phase Full-Bridge Inverter

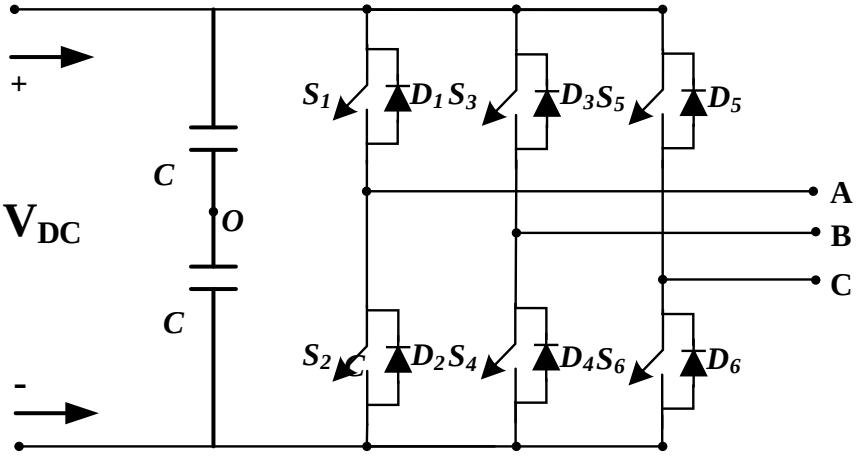


Figure 2.3: Three-Phase Full-Bridge Inverter

The ratio of the inverter's output voltage to its input voltage is defined in inverter applications as the modulation index multiplied by a constant particular to the inverter's structure and control mechanism. By adjusting the modulation index, you may modify the actual output voltage and, therefore, the active and inactive power flow in response to demand. As a result, the input DC link voltage variations affect the control loop, and the inverter's operation must account for these changes to provide reliable and robust service.

Some applications employ an additional DC-DC boost converter step between the power supply and the inverter if the input voltage is insufficient to power the DC-AC inverter. By altering the duty cycle of this converter, it is possible to maintain a constant DC link voltage while relaxing the inverter control that processes the varied input voltage. However, two-stage inverters are less prone to component failure than one-stage converters, resulting in increased cost, decreased power conversion efficiency, and reduced dependability [112].

When utilized in conjunction with the SHEM control method, the changeable DC link voltage at the input of the single-ended inverter has a variable modulation index value, allowing for the selection of alternative switching angles. As a result, it is essential to pre-store the modulation index value in the microcontroller with a set of all these angles for the microcontroller to handle fluctuations in the inverter input DC link voltage. This requirement complicates the modulation index value in discretization, storage, and solution space described later in this chapter.

2.1.1 Railway Applications

Without onboard prime movers or local fuel, electric rail systems are power rail trains and trams. Railways, electric Electric locomotives (which transport passengers or cargo between vehicles) are made up of many electric units (self-powered passenger cars) or a combination of the two. Electricity is often transmitted via a rail network and delivered to trains via big, highly efficient power plants. While some

electric railroads have remarkable power plants and transmission links, most are purchased from power corporations. Railways are often self-sufficient in terms of distribution lines, transformers, and switches. Electricity is usually provided to trains via (nearly) continuous conductors that run down one of two types of railroad tracks. At the track level, it is touched by a sliding "pickup shoe." While run rails are frequently used as return conductors in Gagonson and 3rd rail systems, other systems utilize different 4th rails for this purpose.

Electric trains represented in Figure 2.4 offer significantly higher energy efficiency, cheaper running costs, and lesser pollutants than the primary alternative-diesel engines. Electric locomotives are often quieter, more powerful, more responsive, and more reliable than diesel locomotives. They emit no local pollutants and offer substantial benefits in tunnels and metropolitan areas. Specific electric traction systems incorporate regenerative brakes that return the train's kinetic energy to the supply as electricity to be used on the public utility grid or other trains.

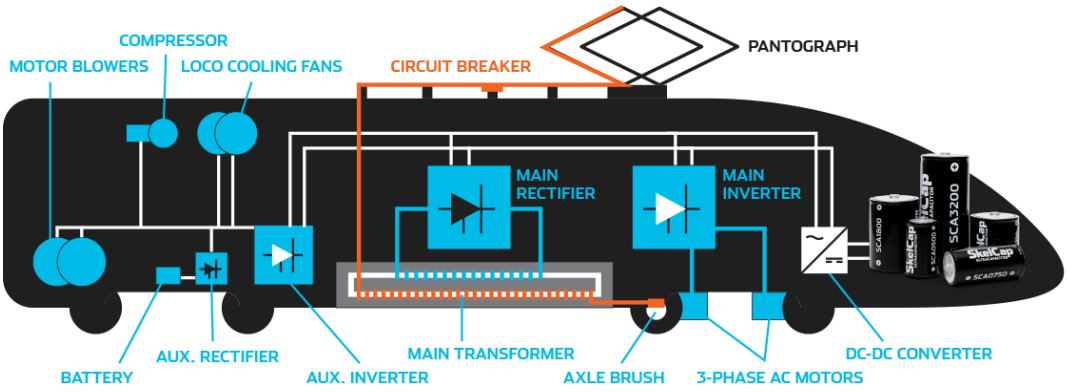


Figure 2.4: Main components of a locomotive [113]

Diesel locomotives also run on electricity supplied by a variety of sources, including solar and wind energy. [114] Throughout history, considerations regarding resource independence have influenced the surprising choice of train lines. Although oil and coal sources are nearly depleted, inland Switzerland, with its abundant hydropower, prompted the network's establishment in reaction to supply shortages during the two World Wars. [115-116] High initial costs can make electric traction uneconomical on low-traffic lines, a relative lack of flexibility (trains require on-track or overhead

cables), and vulnerability to power outages. Electric diesel locomotives and multi-units mitigate some of these issues by operating on diesel power in a power loss or when electricity is available along the route.

If the region is different, the service voltage and frequency may vary, complicating both the service and the locomotive's power. With limited licenses available on overhead electrical cables, efficient double-stack container services can be congested. However, the Indian Railways [117] and the Chinese Railways [118-120] operate double-loaded freight trains on regular and virtual lines.

Rail electrification has grown gradually over the last few decades, and as of 2012, electrified railways accounted for roughly a third of all railroads worldwide. [121] Voltage, current, and contact systems are the three fundamental parameters of electrical systems. Direct current (DC), alternating current (AC), and frequency are all terms used to describe current. Overhead lines, third and fourth rails make up the contact system. The choice of the electronic system is based on the economics of maintenance and capital expenditures compared to revenue from freight and passenger transportation. In the city-to-city area, a variety of techniques are in operation. Certain electric locomotives can operate on a variety of various supply voltages, providing greater operational flexibility.

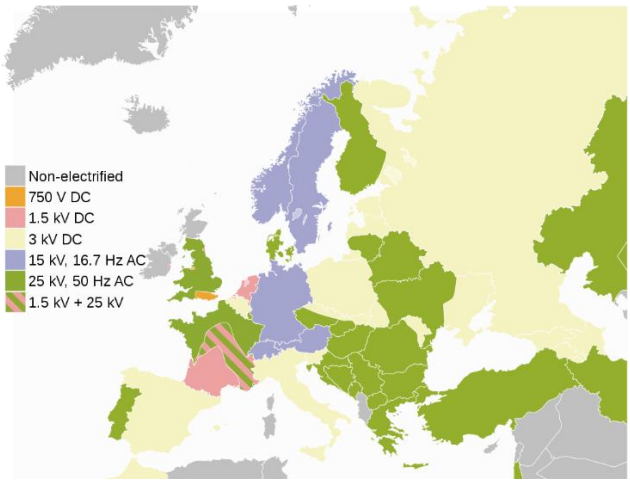


Figure 2.5: European electrical systems: High-speed aircraft from France, Spain, Italy, England, Netherlands, Belgium, and Turkey operate below 25 kV, as well as on high-voltage lines from the former Soviet Union.

It was decided to standardize six of the most widely used voltages to achieve European and international uniformity. For example, a third rail or Gagonson can be used with 750V DC without affecting the rest of the contact system's characteristics. The list of railroad telephone systems includes both standard and non-standard voltage systems and various other voltage systems used in railroad telephone networks around the world. Following the guidelines of the British Standard EN 50163 [122] and the International Electrotechnical Commission (IEC) 60850 [124], the acceptable voltage range for the standard voltage is stated in Table 1. It considers the number of trains that draw current and the distance between the trains and the substation.

Table 2.1: Electrification System Voltage Types and Ratings

System Type	Voltage Levels				
	Min. non-permanent	**Min. permanent**	**Nominal**	**Max. permanent**	**Max. non-permanent**
600 V DC	400 V	400 V	600 V	720 V	800 V
750 V DC	500 V	500 V	750 V	900 V	1000 V
1500 V DC	1000 V	1000 V	1500 V	1800 V	1950 V
3000 V DC	2000 V	2000 V	3000 V	3600 V	3900 V
15 kV AC 16.7 Hz	11 kV	12 kV	15 kV	17.25 kV	18 kV
25 kV AC 50 Hz	17.5 kV	19 kV	25 kV	27.5 kV	29 kV

Most modern electrification systems take AC energy from the power grid, convert it to a lower DC voltage, and rectify it in preparation for use in the locomotive's traction motor. These motors might be direct current (DC) motors or three-phase ac motors that require extra conversion from DC to three-phase ac (using inverters). As a result, both systems confront similar difficulties. In other words, the power grid's high-voltage alternating current is converted and sent to the locomotive's low-voltage direct current. The distinction between alternating current and direct current

electrical systems is in the point at which alternating current is converted to direct current (substations and trains). These will be used on the network, frequently fixed due to existing electrical systems, which infrastructure and energy efficiency expenses have selected.

Electrical energy is lost throughout the conversion and transmission processes. The ohmic losses in wire and power electronics are magnetic field losses in transformers and smoothing reactors (inductors). [123] DC systems are generally used in railway substations due to their compact size and ability to utilize larger, heavier, and more efficient hardware than AC systems converted on trains due to their significantly higher losses. [124] However, the higher voltages employed in many alternating current electrical networks result in lower transmission losses over long distances and the elimination of substations or the employment of more powerful locomotives. Additionally, consider the energy consumed by electronics (including rectifiers), air-cooled transformers, and other conversion hardware. Conventional AC electrical systems operate at significantly greater voltages than standard DC systems. Raising the voltage has the advantage of using less current to transfer a given amount of power (P = V I). By lowering the current, ohmic losses are reduced, allowing for increased separation between lightweight virtual line installations and traction substations without compromising the system's power capacity. On the other side, as the voltage increases, the insulating gap becomes bigger, necessitating the addition of certain infrastructural pieces. Standard frequency alternating current systems can cause supply grid instabilities and require careful planning and design (since power from each substation is drawn in 2 out of 3 phases). Low frequency alternating current systems can be charged via independent power generating and distribution networks or converter substation networks. Additionally, low-frequency transformers, which are found in both substations and automobiles, are quite heavy. Apart from the fact that DC systems are limited in the amount of power they can transmit, there is also the danger of electrochemical corrosion induced by DC currents. [125]

2.2 Discretization of Modulation Index

A feature of digital control is that the SHEM switching angle of the discretized modulation index value implemented as a lookup table for the microcontroller is maintained. For example, [126] employs 50 levels of modulation index values that range from -50MVAr to +50MVAr reactive power control to establish the optimal switching angle value and then stores these modulation index values in a 50x5 look-up table to obtain the optimal switching angle value.

Typically, the modulation index value is obtained in the control loop at the output of the PI controller, which is called the modulation index value. If the value retrieved does not precisely match the value recorded in the lookup table, round it up to the value recorded in the lookup table to ensure that the value recorded in the lookup table is accurate. As a result, the modulation index is set to a value that differs from the value specified in the control loop. The modulated index value was injected into the motor using the conserved switching angle, resulting in the voltage required for the inverter's output and the undeleted subharmonics and other essential voltages. There is excessive active and reactive power and overall harmonic distortion compared to the set limitations. It is possible to show this phenomenon using a basic illustration.

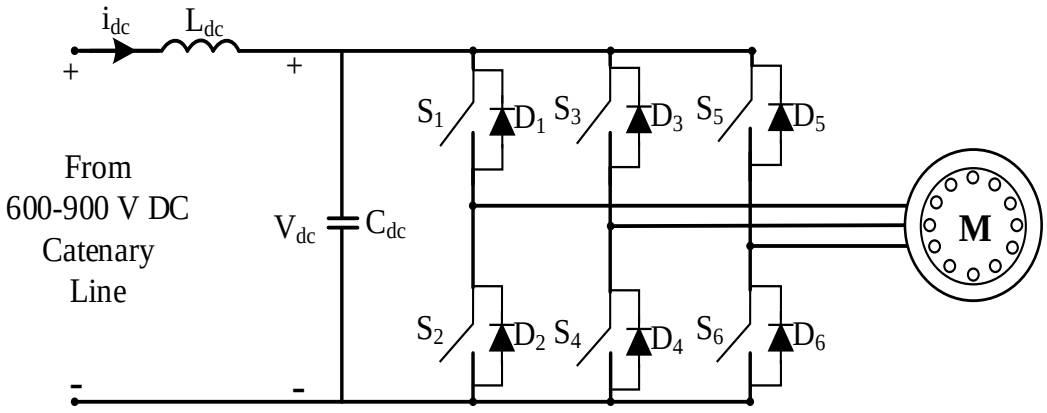


Figure 2.6: Three-phase Two-level Traction Inverter Motor Drive

Figure 2.11 illustrates a three-phase two-level inverter for railway applications. The DC connection voltage fluctuates between 565V and 900V, while the output frequency varies between 50 and 200 Hz. Assume SHEM is used as the control mechanism, with a lookup table of modulation index values and a switching angle to eliminate the fifth, seventh, eleventh, thirteenth, thirteenth, seventeenth, and nineteenth harmonics. Only ten modulation index values are stored. To accomplish this, you will require a file that contains a lookup table for the 10x7 microcontroller switching angles, as seen in Table 2.3.Using the 100 x 7 switching angle look-up table, the error of the active power equal to the worst modulation index error (0.005) is reduced to 118VAr, which corresponds to the position relative to the DC link voltage of 746V. Thus, the modulation index value is greater than the microcontroller value because a look-up table with a more acceptable modulation index value is required due to the microcontroller's failure and the inverter's reactive power flow. As a result, substantial storage space must be allocated.

Table 2.2: Sample Look-up Table Including Switching Angles According to Modulation Index

Modulation index	**Switching angles (Degree)**						
m	α_1	α_2	α_3	α_4	α_5	α_6	α_7
0.10	α_{11}	α_{12}	α_{13}	α_{14}	α_{15}	α_{16}	α_{17}
0.20	α_{21}	α_{22}	α_{23}	α_{24}	α_{25}	α_{26}	α_{27}
0.30	α_{31}	α_{32}	α_{33}	α_{34}	α_{35}	α_{36}	α_{37}
0.40	α_{41}	α_{42}	α_{43}	α_{44}	α_{45}	α_{46}	α_{47}
0.50	α_{51}	α_{52}	α_{53}	α_{54}	α_{55}	α_{56}	α_{57}
0.60	α_{61}	α_{62}	α_{63}	α_{64}	α_{65}	α_{66}	α_{67}
0.70	α_{71}	α_{72}	α_{73}	α_{74}	α_{75}	α_{76}	α_{77}
0.80	α_{81}	α_{82}	α_{83}	α_{84}	α_{85}	α_{86}	α_{87}
0.90	α_{91}	α_{92}	α_{93}	α_{94}	α_{95}	α_{96}	α_{97}
1.00	α_{101}	α_{102}	α_{103}	α_{104}	α_{105}	α_{106}	α_{107}

As mentioned previously, a similar strategy can be used with the multi-level inverter. If the DC link voltages of the various H-bridges are not identical, the SHEM equations for all H-bridges are adjusted [33]. This is not the case for a two-level inverter. There is no SHEM equation DC link voltage term to investigate in the following chapter, save to manage the primary voltage, as mentioned in the following chapter.

As a result, changing the switching angles affects just the base and reactive voltages. When multi-level inverters are used, rounding the modulation exponent does not reduce lower-order harmonics, as the exponent value affects the entire equation set. As a result, the reactive motor power is increased, as is the total harmonic distortion.

2.3 Lookup Table Storage

In practice, a set of ten or more switching angles is recorded to provide a more continuous range of modulation index values. However, digital control features can be stored in the microcontroller in a finite number of sets of angles. Saving a more significant number of angles reduced the inaccuracy introduced by discretization, but this requires a larger microcontroller. For instance, a microcontroller's switching angle resolution is 100x7, yet maintaining a 32-bit-defined 10x7 look-up table in Table 2.3 consumes 280 bytes of RAM. The look-up table is saved in 2.73 kB of memory.

Additionally, one or more switching angles must be included in any combination of angles to remove one or more harmonics, resulting in a table size of 100x8. This increase consumes around 400 bytes more memory from the microcontroller. The size of the look-up table will be more significant if it contains more discretization modulation index values and switching angles to eliminate lower-order harmonics.

The advantage of the look-up table over the offline program is that the SHEM online application does not require the use of a look-up table to calculate switching angles. The additional computational load offsets this gain. Nevertheless, when SHEM's

evolutionary navigation algorithm is used live, this computing load does not scale linearly with the number of harmonics removed. As detailed in the following chapter, metaheuristic algorithms can solve the corresponding equations generated by the parallel search idea people implement in a population. The more harmonics removed, the greater the computational load for determining the switching angle set's goodness-of-fit value. However, the population of evolutionary navigation algorithms does not have a linear rise in the number of objects. Even with the same number of things, careful algorithm parameters can answer a higher-order set of equations.

2.4 Infeasible Region of Solution Space

SHEM is a non-linear transcendental equation with sinusoidal terms for two- and multi-level inverters. The equations for switching angles that produce the desired modulation index and eliminate the given number of harmonics are resolved. The modulation index has a value between 0 and 1, and within the majority of this range, you can find a solution that satisfies the restrictions. There is, however, a limited area inside this range where no feasible solutions exist. That is, no set of switching angles exists that satisfies all restrictions.

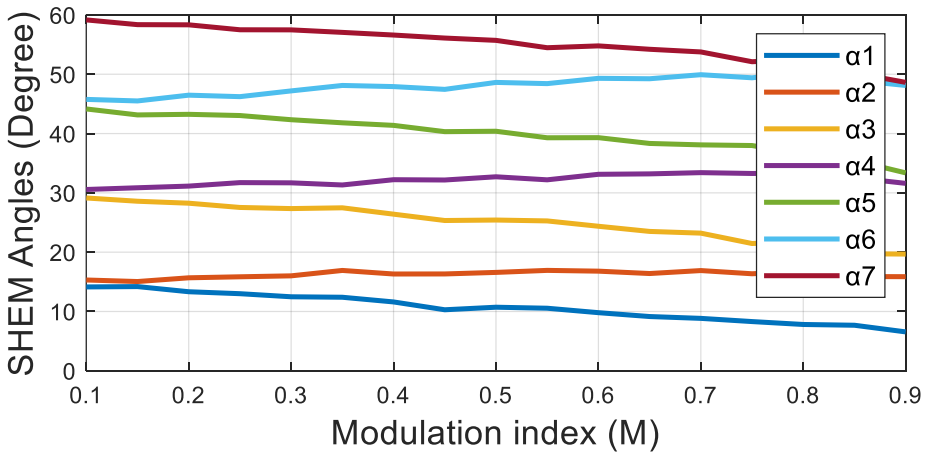


Figure 2.7: Switching Angles versus Modulation Index Distribution of the VSI

When the 5th, 7th, 11th, 13th, 17th, and 19th harmonics are removed from the output line, the neutral voltage of a two-level three-phase inverter controlled by seven SHEM is switching angles. This location is depicted well in Figure 2.12. Following a 0.9 modulation index, some switching angles converge, and the voltage waveform at the inverter's output indicates that the notch is discarded. Additionally, by examining the harmonic content of this waveform, we can determine that it is not erased, as illustrated in Figure 2.12. If SHEM is to be applied online, the modulation index is limited at the region's boundaries, resulting in a set of multiple new switching angles for infeasible values. This section continues with the solution space to be between modulation index 0.1 and modulation index 0.9. If a look-up table is used, the missing solution space can be filled using the angles discovered for other issue configurations. Rather than minimizing individual harmonics, it can be used as a parameter to reduce the total harmonic distortion of the inverter's output voltage. However, because this implies that the features of the viable region vary at its boundaries, the transition angle must be carefully determined and handled to ensure a seamless transition. Metaheuristic algorithms have several significant advantages over present iterative algorithms. We thoroughly search the solution spaces of these algorithms, which will be discussed in further detail in the following chapter, and avoid becoming caught in local minima.

While conventional iterative algorithms require good initial points to converge, evolutionary algorithms may typically identify a set of solutions even when the initial population is arbitrary. The iterative method fails and recommends a set of non-viable solution space transition angles. Nevertheless, the evolutionary process can discover a collection of solutions in these fields. This solution does not eliminate the required harmonics due to the impossibility of doing so. Still, it is the best set of solutions that minimizes these harmonics and is suitable for the provided modulation index. It ensures consistency and a seamless transition inside the solution space. By carefully configuring the metaheuristic algorithm settings, you can find a solution set considerably faster than you can with an iterative method, allowing you to use SHEM online.

CHAPTER 3

META-HEURISTIC OPTIMIZATION ALGORITHMS APPLIED TO SHE APPLICATIONS

This chapter will discuss metaheuristic approaches in detail. The application of SHE to a three-phase voltage source inverter will be discussed, as well as the metaheuristic approaches used to optimize it, which include particle swarm optimization (PSO), gray wolf optimization (GWO), whale optimizer algorithm (WOA), and harris hawks optimization (HHO). Additionally, a comparison of these techniques will be offered.

3.1 Meta-Heuristic Optimization Algorithms

In today's very competitive world, people are trying to take advantage of the highest yield or to benefit from a limited amount of resources. For example, the fundamental purpose of engineering design is to conform with basic standards and create good economic outcomes by selecting design variables that meet all design requirements and the lowest potential costs.

Optimization offers a method to solve this type of problem. The phrase optimization relates to the analysis of issues. One tries by carefully selecting the values of variables in/within an allowable set to minimize or maximize a function. A wide range of research in this field of knowledge has been undertaken in the hope of developing practical and effective optimization algorithms. On the other hand, several studies have also focused on using current algorithms in practical projects.

In the past, gradient-oriented algorithms were usually employed to search for solution space at an initial starting point using gradient-oriented information[127-128]. In general, gradient-based algorithms converge more quickly and can achieve more great precision answers than stochastic alternatives. However, it can be costly

or perhaps impossible to find the minimum to gain gradient information. Furthermore, algorithms of this type can only converge to local minimum values. In addition, the effectiveness of these strategies requires an appropriate beginning point. Prohibited areas, side limits, and non-smooth or non-convex functions should be considered for many optimization problems. Through these methods, it is therefore not easy to resolve these non-convex optimization problems.

On the other hand, other optimization methods are not confined in the way mentioned above, called metaheuristic algorithms. Their potential to explore and locate good locations in the search area at affordable computer time makes them suited for worldwide searches. For most optimization tasks, metaheuristic algorithms are often successful [129-130]. Its successful application in a wide range of engineering, physics, chemistry, art, economy, marketing, genetics, research on operations, robotics, social sciences, and policy is demonstrated because these solutions do not simplify or assume the original issue. The word heuristic is derived from ancient Greek words, meaning that new tactics (Rules) are to be discovered to resolve difficulties. The meta suffix also signifies "high-quality methodology" and is also a Greek term. Glover introduced in the study the term metaheuristic[131].

A heuristic technique may be seen as a procedure that can find, but not necessarily an optimal solution to the given particular problem, a very excellent workable answer. The quality of the solution achieved cannot be guaranteed, but a well-conceived heuristic method may usually produce an almost optimal solution. The technique should also be efficient enough for substantial problems to be addressed. The heuristic approaches are generally seen as the iterative algorithm, where each iteration involves the search for a new solution better than previously obtained. The provided answer is the best one obtained during any iteration after a reasonable time when the algorithm is terminated. A metaheuristic is described as an iterative process for generating a subordinate heuristic through the intelligent combination of multiple concepts for exploration (world searcher) and using (locally searched) search areas.

In various applied math, engineering, medicine, economics, and other sciences, metaheuristic algorithms are found in many ways. These methods are widely used in designing numerous civil, mechanical, electrical, and industrial systems. In addition, the continuous emphasis on the interdisciplinary nature of the discipline is one of the most critical developments in optimization.

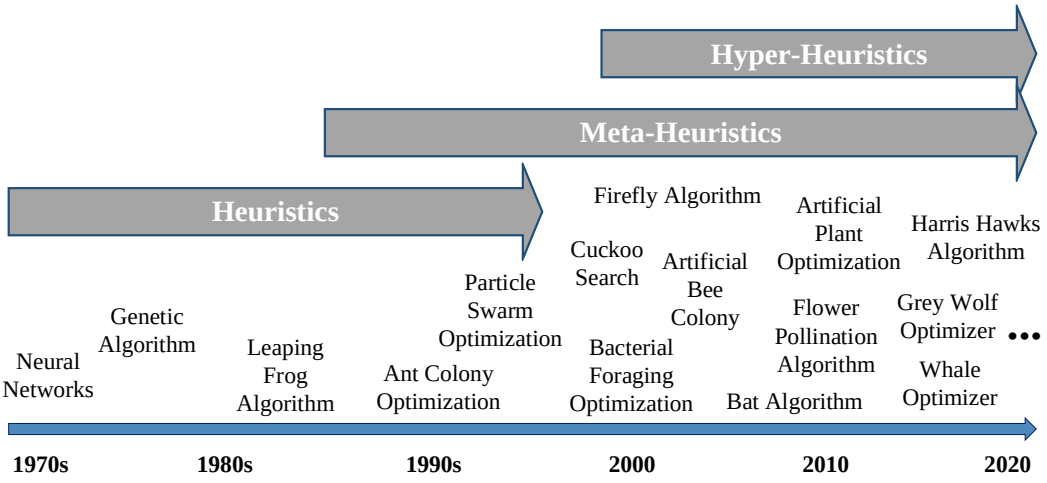


Figure 3.1: Development of Bio-motivated Calculations

The metaheuristic addressed in Figure 3.1 is viewed as an algorithmic design that, for the most part, applies to different issues with improvement, with a couple of changes to adjust to the given circumstance. Metaheuristics additionally have fundamental attributes that can be summed up as:

- Broad applicability: they could be applied as problems of function optimization to any issue which can be formulated.
- Towards hybridization: can be merged with more traditional techniques of optimization.
- Easy to use: usually simpler to understand and apply.
- Efficiency and flexibility: problems can be solved more quickly. Moreover, they are very flexible and are easy to design and implement.
- Because of the internal complexity, meta-heuristics could be validated problem that prevents exact techniques from being used and (ii)

considerable problem amount of possible solutions to ensure that exhaustive algorithms are not used.

However, the meta-heuristic algorithms should be noted here as certain disadvantages:

- In general, the performance of optimization depends significantly on the tuning of acceptable parameters.
- When compared with more traditional techniques, they have no mathematical "sound" basis.
- At the optimality, they can't prove.
- They may not reduce the search space.
- The repetitiveness of optimization results achieved under the same settings for the initial condition is not ensured.

3.2 Particle Swarm Optimization (PSO)

Particle swarms originated as a simulation of a reduced social structure. The original objective was to depict an elegant yet surprising new herd choreography visually. Initial simulations were adjusted to incorporate closest neighbor velocity matching, eliminate auxiliary variables, and integrate distance-based multidimensional search and acceleration (Kennedy and Eberhart 1995 and Kennedy 1995). During the algorithm's evolution, it can be understood that the conceptual model was indeed an optimization tool. The algorithm has eliminated numerous parameters unrelated to optimization to provide a straightforward original implementation through trial and error (Eberhart, Simpson and Dobbins 1996) [132]. The PSO, like the Genetic Algorithm (GA), is initialized with a random solution population. However, it is distinct from GA in that each possible solution is given an arbitrary velocity and the resulting solution of particles "flows" through the problem space.

Each particle maintains a map of the problem space's coordinates associated with the ideal solution (goodness of fit) obtained thus far. (Also stored is the fitness value.)

This value is referred to as pbest. Another "best" value to track in the global version of the Particle Flock Optimization software is the population's overall maximum value and position as determined by all particles in the population thus far. This site is referred to as gbest. Particle family optimization is a concept that entails adjusting (accelerating) the velocity of each particle at each time step in order to move it closer to its pbest and gbest positions (global version of PSO). Acceleration is weighted in random ways, and distinct random numbers are generated for the pbest and gbest places. Additionally, there is a local version of the PSO. Along with the pbest, each particle in this variant tracks the best solution, denoted by lbest, obtained near the particle's local topological curl [133].

For implementing a global version of PSO, the following is the (original) approach to follow:

1. To begin, create a population (array) of particles with arbitrary positions and velocities in the d dimension of the problem space in the d dimension of the problem space.
2. Evaluate the desired optimization fitness function for the d variable for each particle.
3. Compare the suitability assessment of particles to the suitability assessment of particle pbest. When used to a better-than-pbest condition, this function sets pbest's current value and its location in d-dimensional space to the current values and positions of the best and worst values, respectively.
4. Compare the fitness rating to the best overall rating before the population is taken into consideration.
5. If the current value is better than the previous best value, it resets the previous best value to the array index value of the current particle.
6. Modify the particle's velocity and position following equations (3.1) and (3.2), respectively.

$$v_{id} = v_{id} + c_1 * rand() * (p_{id} - x_{id}) + c_2 * rand() * \left(p_{gd} - x_{id}\right) \quad (3.1)$$

$$x_{id} = x_{id} + v_{id} \quad (3.2)$$

7. Repeat step (2) until a criterion is met, usually a good enough goodness of fit or the maximum number of iterations (generations).

The maximum velocity Vmax of each dimensions particle is confined to the velocity of the other-dimensional particles. The sum of the accelerations and the speed in that dimension surpasses the maximum velocity (v_{max}). The user supplied this parameter and is limited by the maximum velocity v_{max} of that dimension. As a result, v_{max} is an essential parameter. It specifies the level of precision or granularity with which the search area between the current location and the target location is searched for (best so far). If v_{max} is set too high, the particles may be able to travel past a stable solution.

On the other hand, if v_{max} is set too low, the particles may not be able to migrate far enough outside of the locally good region to be effective. In actuality, they may be locally ideally constrained, unable to travel for a long enough period to reach a better location in the problem space, resulting in a halt in their progress. For Equation 3.1, the acceleration constants c_1 and c_2 are defined as the weight of the probabilistic acceleration protest that pushes each particle to the best possible position (*pbest*) and the best possible position (*gbest*). As a result, changing the values of these constants alters the degree of "stress" in the system. Using lower numbers allows particles to wander away from the target area before being drawn back in. Higher values represent movements towards or through the target area that are more abrupt. Early experience (in most cases, trial and error) with particle population optimization determined that the acceleration constants c_1 and c_2 should be 2.0 and 2.0, respectively, in virtually all cases. As a result, Vmax is the sole parameter that we tweak regularly, and it is typically set at approximately 10-20 percent of the dynamic range of the variable in each dimension.

A 'local' form of the particle family was designed in particular based on the results of social simulations, which was the focus of this project. In this version, the particle only has the most up-to-date information on itself and its neighbors

rather than about the entire group. However, rather than traveling towards a form of stochastic average of *pbest* and *gbest* (the best location in the entire group), the particle goes towards the point specified by pbest and "*lbest*." neighborhood.

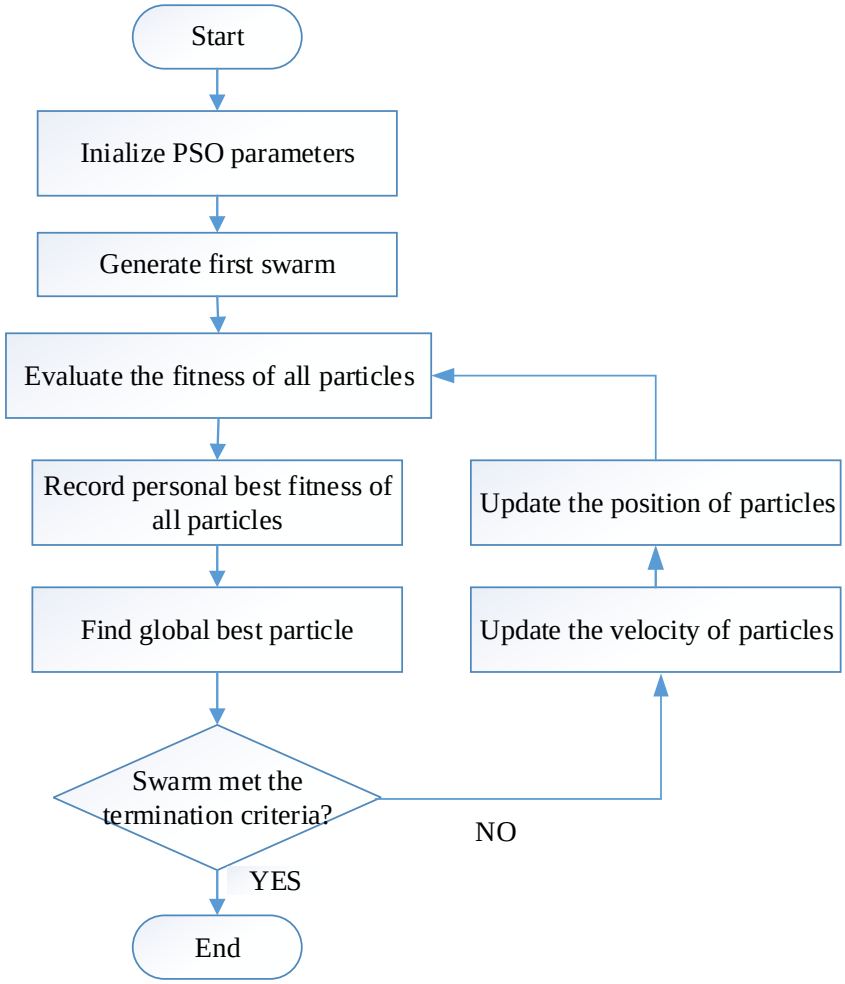


Figure 3.2: Basic Flowchart of PSO

Example: If the neighborhood size is defined by 2, particle I compare the goodness-of-fit value to particle (i-1) and particle I + 1), and the neighborhood size is determined by 2. Topology neighbors are the closest neighbors in a topology. During the run, there is no change in the neighborhood or with the neighbors. The only difference between the global and neighborhood versions of the process established in step 6 is that the global best position pid in Equation 3.1 is replaced with the neighborhood best position *pid*. Because of early experience (primarily via trial and

error), approximately 15 percent of the population was utilized in various applications. Because of this, it is not rare for a population of 40 particles to have six or three topological neighbors on either side of them.

The problem determines the population size that is chosen. The population size of 20-50 people is likely to be the most prevalent. For other evolutionary algorithms (such as genetic algorithms and evolutionary programming), the total number of ratings required for optimal PSO results represented in Figure 3.2 for populations smaller than those commonly used is greater than the number of ratings. The ratings are necessary for optimal PSO results for populations more prominent than those widely used (population size multiplication generation).

3.2.1 Inertial weight

Specifically, the maximum velocity Vmax serves as a limit that controls the swarm of particles' ability to navigate about the world. As previously said, the larger the v_{max}, the easier it is to navigate on a global scale, and the lower the v_{max}, the easier it is to navigate on a local scale. The concept of intertia weight was invented to have greater control over navigation and development processes. The goal was to eliminate the requirement for v_{max} as a result of this research. According to the published literature, inertial weights in particle swarm optimization methods were first reported in 1998. Specifically, Equations 3.3 and 3.4 explain the velocity and position update equations that take into account inertial weights. It can be observed that this equation is the same as Equations 3.1 and 3.2, except that the inertia weight *w* is added as a multiplication factor of *vid* in Equation 3.1, and w is added as a multiplication factor of *vid* in Equation 3.2 and 3.3. Many applications benefit from the usage of inertial weight w, which improves their overall performance. During execution, the initially developed w frequently drops linearly from approximately 0.9 to 0.4. It is possible to balance global and regional navigation and usage by carefully selecting inertial weights. This reduces the average number of iterations required to obtain a sufficiently optimal solution.

$$v_{id} = v_{id} + c_1 * rand() * (p_{id} - x_{id}) + c_2 * rand() * (p_{gd} - x_{id}) \quad (3.3)$$

$$x_{id} = x_{id} + v_{id} \quad (3.4)$$

3.2.2 Coefficient of Constriction

Because particle-family optimization is a result of efforts to model social systems, the whole mathematical foundation of the algorithm and the methodology have not yet been defined in its current form. There have been several attempts to begin laying the groundwork for this foundation during the last few years. Clerc (1999) conducted a recent study in which he discovered that the usage of constriction coefficients might be required to assure the convergence of particle-group algorithms. This white paper does not offer a complete discussion of the shrinkage coefficient; nevertheless, Equation 3.5 illustrates a simple way to include the shrinkage coefficient by assuming that *K* is a function of c_1 and c_2, as indicated in Equation 3.6.

$$v_{id} = K * [v_{id} + c_1 * rand() * (p_{id} - x_{id}) + c_2 * rand() * (p_{gd} - x_{id})] \quad (3.5)$$

$$K = \frac{2}{|2-\varphi-\sqrt{\varphi^2-4\varphi}|}, \text{ where } \varphi = c_1 + c_2,\ \varphi > 4 \quad (3.6)$$

By convention, when the Clerc contraction method is used, it is set to 4.1, implying that the constant multiplier *K* is 0.729. As a result, multiply the initial velocity by 0.729 and the two (p-x) terms by 0.729 * 2.05 = 1.49445. (random multiplication between 0 and 1). Because v_{max} was superfluous while employing Clerc's contraction approach, it set to 100,000 for early studies and applications. However, a more appropriate system to apply as a "rule of thumb" in later investigations and applications (Eberhart and Shi 2000) is to restrict v_{max} to x_{max}, the dynamic range of each variable in each dimension, and to select *w*.., c_1 and c_2 are determined using Equations 3.5 and 3.6. Numerous evolutionary algorithms are applicable to the solution of static problems. However, the condition of many real-world systems changes regularly (or continuously). These system state changes frequently necessitate near-constant optimization. The application of particle swarm

optimization to tracking and optimizing dynamic methods has been proved to be successful.

3.3 Grey Wolf Optimization (GWO)

The grey wolf (Canis lupus) is a member of the Canidae family. Grey wolves are apex predators, which means they are at the top of the food chain. Grey wolves, for the most part, prefer to live in packs [134]. The average group size is between 5 and 12. What makes them unique is that they have a relatively rigid social dominance hierarchy, as illustrated in Figure 3.3.

The leaders, referred to as alphas, are a male and a female. The alpha is primarily responsible for making decisions on hunting, sleeping location, and wake-up time. The pack dictates the alpha's decisions. However, democratic behavior has been observed in which an alpha wolf follows the pack's other wolves. The entire group recognizes the alpha during meetings by holding their tails down. The alpha wolf is also referred to as the dominant wolf, as the pack must obey their commands [46]. Alpha wolves are not permitted to mate outside the group. Interestingly, the alpha is not often the most vital member of the pack but is the most adept at managing it. This demonstrates that a pack's organization and discipline are far more crucial than its strength.

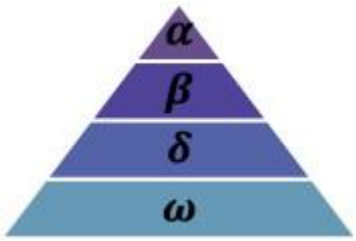


Figure 3.3: The Hierarchy of Wolves

Beta is the second rank in the grey wolf hierarchy. Betas are subordinate wolves who assist the alpha with decision-making and group activities. The beta wolf can be male or female, and he or she is probably the most significant prospect to take over as an

alpha wolf if one of the alpha wolves dies or becomes very old. While the beta wolf should revere the alpha, they should also command the other lower-level wolves. It acts as an advisor to the alpha and as a disciplinarian for the pack. Throughout the group, the beta reinforces the alpha's directives and provides feedback to the alpha.

Omega is the grey wolf with the lowest ranking. The omega serves as a scapegoat. Omega wolves must always yield to all other dominant wolves. They are the last wolves permitted to hunt. While the omega may be a minor member of the pack, it has been noted that when the omega is lost, the entire group has internal conflict and issues. This is because the omega is channeling the violence and frustration of all wolves (s). This contributes to the satisfaction of the entire pack and the maintenance of the pack's dominance structure. In some circumstances, the omega is also the pack's babysitter.

When a wolf is not an alpha, beta, or omega, they are referred to as a subordinate wolf (or delta in some references). Although delta wolves must bow to alphas and betas, they rule the omega. This category includes scouts, sentinels, elders, hunters, and caregivers. Scouts are responsible for monitoring the territory's limits and alerting the pack to any threat. Sentinels protect and ensure the pack's safety. Elders are seasoned wolves who were once alpha or beta. When hunting prey and supplying food for the group, hunters assist the alphas and betas. Finally, caregivers are accountable for the pack's weak, unwell, and injured wolves.

Along with the wolves' social structure, collective hunting is another fascinating social characteristic of grey wolves. Outline the following phases of grey wolf hunting:

- Tracking, pursuing, and closing in on the prey.
- Pursue, encircle, and annoy the victim until it comes to a halt.
- Attack the prey.

3.3.1 Social hierarchy

Gray wolves are members of the canine family and are regarded as the food chain's top predators. Often, gray wolves prefer to live in packs. Social hierarchies such as alpha, beta, delta, and omega are of particular importance. We consider alpha (α) to be the optimal solution among wolves when developing a GWO to mathematically simulate the social order of gray wolves. As a result, the second and third-best solutions are denoted by the letters beta (β) and delta (δ). Omega (ω) is assumed for the remaining possible solutions. The wolf determines the bait (optimization) in the GWO algorithm then pursues these three wolves.

3.3.2 Surrounding prey

Along with social hierarchy hypotheses, another social behavior is group hunting for grey wolves. Grey wolves hunt in three stages: (1) following, chasing, and bringing the hunt closer; (2) pursuing, hunting, and stalking its prey until it stops; and (3) direct attack on the hunt.

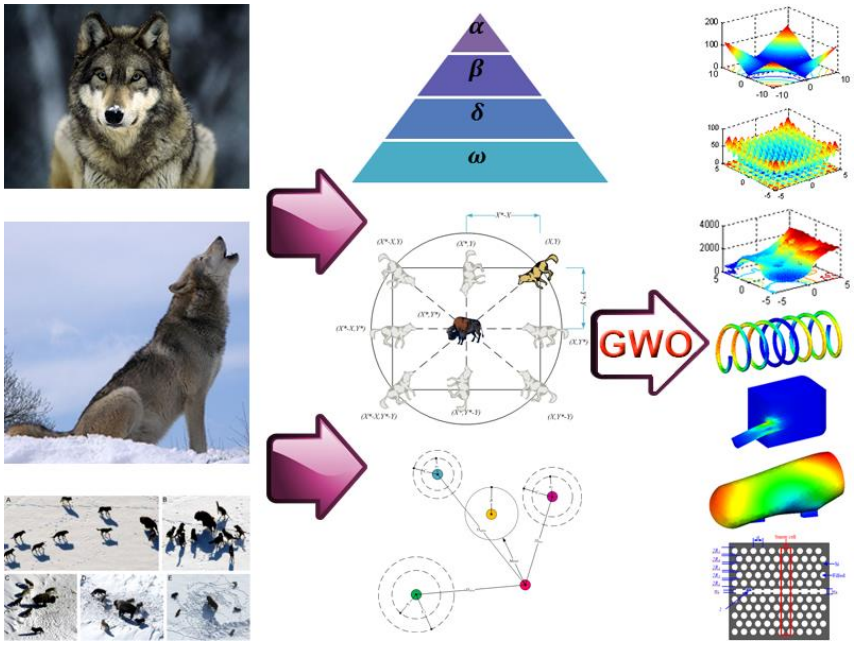


Figure 3.4: A Summary Form of GWO

Mathematically the model containment behavior is explained in the following equations:

$$\vec{D} = |\vec{C} \cdot \vec{X}_p(t) - \vec{X}(t)| \tag{3.7}$$

$$\vec{X}(t+1) = \vec{X}_p(t) - \vec{A} \cdot \vec{D} \tag{3.8}$$

In Equations 3.7 and 3.8 equations t denotes iteration, A and C are coefficient vectors; $\vec{X}_p$ is the position vector of the prey and $\vec{X}$ defines the position vector of a grey wolf.

$$\vec{A} = 2\vec{a} \cdot \vec{r_1} - \vec{a} \tag{3.9}$$

$$\vec{C} = 2 \cdot \vec{r_2} \tag{3.10}$$

a decreases with iterations 2 to 0 values in Equation 3.9. r_1 and r_2 values randomly change in [0, 1] range, and similarly, A takes random values in [$-a$, a] range.

3.3.3 Hunting

Grey wolves are primarily interested in alpha, beta, and delta positions. They disperse to search for and approach their victim. When *A* is higher than 1 or less than -1, the model deviates mathematically. Apart from that, discovery is possible, and GWO functions appropriately on a global scale. As a result, | *A* |>1 keeps wolves away from the prey.

C is another component of GWO that facilitates exploration. *C* is a vector containing random values in the range [0, 2]. These components are random coefficients representing the equation distance associated with determining the prey effect at (C_1) or *C* intervals. It enables GWO to maintain control over unexpected behavior, discovery, and local preference throughout the optimization process. It's worth noting that *C* does not drop linearly compared to *A*. GWO instructs *C* to take random values during the initial and last iterations to locate the prey. This condition of local optimum is particularly advantageous in recent iterations.

3.3.4 Attacking prey (exploitation)

When approaching prey, the a value lowers linearly to model mathematically. As a result, the value *A* is transformed into a random value in the range [-a, a]. GWO drives worms to attack prey when *A* random values are in the range [-1, 1] (|*A* < 1).

3.3.5 Search for prey (exploration)

To mathematically imitate gray wolf hunting behavior, the top three best solutions ever obtained are recorded and push other search agents (including's) to update their positions to match the best search agents' positions. The following formulas define the best positions of agents:

$$\overrightarrow{D_\alpha} = |\overrightarrow{C_1} \cdot \vec{X}_\alpha - \vec{X},\ \overrightarrow{D_\beta} = |\overrightarrow{C_2} \cdot \vec{X}_\beta - \vec{X}|, \overrightarrow{D_\delta} = |\overrightarrow{C_3} \cdot \vec{X}_\delta - \vec{X}| \tag{3.11}$$

$$\overrightarrow{X_1} = \vec{X}_\alpha - \overrightarrow{A_1} \cdot \overrightarrow{D_\alpha},\ \ \overrightarrow{X_2} = \vec{X}_\beta - \overrightarrow{A_2} \cdot \overrightarrow{D_\beta},\ \overrightarrow{X_3} = \vec{X}_\delta - \overrightarrow{A_3} \cdot \overrightarrow{D_\delta} \tag{3.12}$$

$$\overrightarrow{X(t+1)} = \frac{X_1 + X_2 + X_3}{3} \tag{3.13}$$

To summarize, the quest for gray wolves in GWO begins at random. The worms estimate the prey's likely location based on the positions of the alpha, beta, and delta worms. Each candidate's answer modifies the hunt's distance. This parameter lowers indefinitely from 2 to 0. Candidate solutions will assist you in navigating the search. Finally, if the final conditions are sufficient, the GWO process is ended. Finally, Figure 3.5 depicts the GWO flowchart.

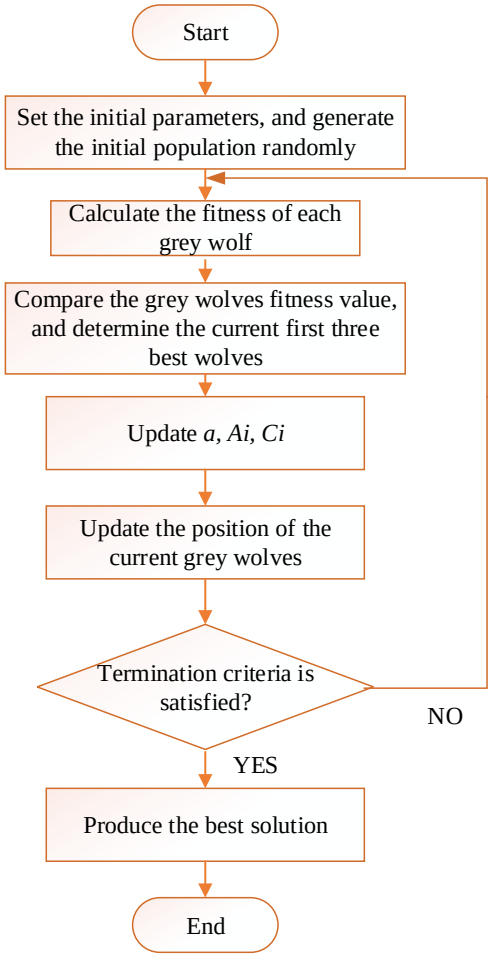


Figure 3.5: The Flowchart of the GWO Algorithm

3.4 Whale Optimization Algorithm (WOA)

Whales are awe-inspiring creatures. They are regarded to be the world's largest mammals. Adult whales can reach a length of 30 meters and a weight of 180 tons. There are seven distinct species of this colossal mammal: the killer, the Minke, the Sei, the humpback, the right, the finback, and the blue. Whales are primarily thought of as predators. They never sleep, as they must breathe from the ocean's surface. Indeed, half of the brain is dormant. What's fascinating about whales is that they are regarded as brilliant animals capable of emotion [135].

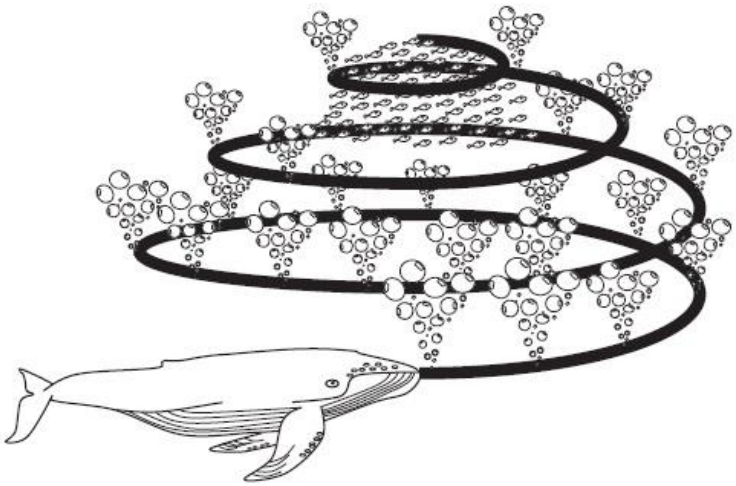


Figure 3.6: Bubble-net Feeding Behavior of Humpback Whales.

Whales share spindle cells with humans in specific parts of their brains. In humans, these cells are involved in judgment, emotion, and social interaction. In other words, the spindle cells distinguish us from other animals. Whales have twice the number of these cells as adult humans, which is one of the primary reasons for their intelligence. It has been demonstrated that whales can think, learn, judge, communicate, and even become emotional the same way as humans do, albeit with a far lower degree of intelligence. Whales (most notably killer whales) have also been documented to develop their own vernacular.

Another fascinating aspect is whales' social behavior. They either live alone or in small groups. They are, however, typically observed in groups. Certain species (for example, killer whales) can live in families throughout their lives. The humpback whale is one of the giant baleen whales (Megaptera novaeangliae). Adult humpback whales are around the size of a school bus. Their preferred prey items are krill and tiny fish herds. This mammal is seen in Figure 3.6.

3.4.1 Encircling prey

Humpback whales have an instinctual sense of where prey is and will encircle it if they detect it. Because the location of the objective prey is unknown from the

previous, the WOA calculation assumes that the current best applicant arrangement is the accurate prey or a close approximation of the ideal. The recognized best pursuit operator begins to emphasize their position, and the other inquiry experts make an effort to refresh their places concerning the best hunt operator. The following conditions demonstrate this pattern of behavior:

$$\vec{D} = |\overrightarrow{C} \cdot \vec{D}^{*}(t) - \vec{X}(t)| \tag{3.14}$$

$$\vec{X}(t+1) = |\overrightarrow{C} \cdot \vec{D}^{*}(t) - \vec{A} \cdot \vec{D}| \tag{3.15}$$

where *t* is the current iteration, *A* and *C* denote coefficient vectors, *X* denotes the position vector of the best solution acquired thus far, and X denotes the worst solution obtained thus far. *X* should be adjusted in each iteration if a better solution is discovered. It should be stressed throughout the rest of the document.

The following is the formula for calculating the vectors *A* and *C*:

$$\vec{A} = 2\vec{a} \cdot \vec{r} - \vec{a} \tag{3.16}$$

$$\vec{C} = 2 \cdot \vec{r} \tag{3.17}$$

where an is lowered linearly from 2 to 0 over iterations (in both the exploration and exploitation phases), and *r* is a random vector in the range [0,1].

The *A* and *C* vectors are critical in determining the optimal position of the best agent. It should be emphasized that by defining the random vector r, any place in the search space can be reached. As a result, Equation 3.15 enables any search agent to change its position relative to the current best solution, simulating circling the prey. As discussed previously, humpback whales also use the bubble-net approach to assault their prey. Mathematically, this technique is as follows:

3.4.2 Bubble-net attacking method (exploitation phase)

With the express objective of scientifically demonstrating the humpback whales' bubble-net assault technique, two techniques are presented as follows:

3.4.2.1 Shrinking encircling mechanism

It is performed by lowering the value of an in Equation 3.16. The fluctuation range of *A* is likewise reduced by a. In other words, *A* is a random value in the interval [-*a*, *a*], where an is decreased from 2 to 0 between iterations. By using random values for *A* in the range [-1,1], the new position of a search agent can be defined as any point between the agent's initial position and the position of the current best agent.

3.4.2.2 Spiral updating position

The following spiral equation is then developed between the position of the whale and the position of the prey to simulate the helix-shaped movement of humpback whales:

$$\vec{X}(t+1) = \overrightarrow{D'} \cdot e^{bl} \cdot \cos(2\pi l) + \vec{X}^*(t) \tag{3.18}$$

Where ($\overrightarrow{D'}$)= $\vec{X}(t+1)$- $\overrightarrow{X^*}(t)$ and represents the distance between the l th whale and the prey (best answer obtained thus far), b is a constant used to define the shape of the logarithmic spiral, l is a random number in the range [-1,1], and it is a component-by-component multiplication. Note that humpback whales move simultaneously around their prey in a diminishing circle and along a spiral-shaped course. We assume a 50% chance of using either the shrinking encircling mechanism or the spiral model to update the whales' positions throughout optimization to describe this concurrent behavior. The following is the scientific model:

$$\vec{X}(t+1) = \begin{cases} \vec{X}^*(t) - \vec{A} \cdot \vec{D} & p < 0.5 \\ \overrightarrow{D'} \cdot e^{bl} \cdot \cos(2\pi l) + \vec{X}^*(t) & p \leq 0.5 \end{cases} \tag{3.19}$$

where p is a random number between 0 and 1. Regardless of the bubble-net technology, humpback whales arbitrarily hunt for prey. The following is the scientific model for the search.

3.4.3 Search for prey (exploration phase)

Given the *A* vector variation (exploration), a similar strategy can be used to search for prey, given the A vector variation (exploration). Indeed, humpback whales search randomly based on their proximity to one another. As a result, we employ A with arbitrary values greater than or equal to 1 to push the search agent to move away from a reference whale. In contrast to the exploitation phase, the exploration phase updates the position of a search agent using an arbitrarily chosen search agent rather than the best search agent found thus far. Together with (|A|) > 1, this approach emphasizes exploration and enables the WOA algorithm to do a global search. The following is the scientific model:

$$\vec{D} = |\overrightarrow{C} \cdot \vec{X}_{rand} - \vec{X}| \tag{3.20}$$

$$\vec{X}(t+1) = |\vec{X}_{rand} - \vec{A} \cdot \vec{D}| \tag{3.21}$$

where $\overrightarrow{X_{rand}}$ is a randomly generated position vector (a randomly selected whale from the current population). The WOA algorithm begins with a collection of randomly generated solutions. At the end of each iteration, search agents update their positions about either a randomly chosen search agent or the best solution thus far achieved. The parameter is reduced from 2 to 0 to enable exploration and exploitation. When |(*A*)| >1, a random search agent is picked for updating the position of the search agents, whereas the optimal solution is chosen when |(*A*)|<1, WOA can move in a spiral or circular fashion depending on the value of *p*. Finally, the WOA algorithm is ended when a termination requirement is satisfied.

Figure 3.7 illustrates the WOA algorithm's pseudocode. WOA can be considered a global optimizer from a purely hypothetical standpoint due to its exploration/exploitation capability. Additionally, the suggested hyper-cube mechanism defines a search region around the best solution and enables other search agents to utilize the domain's current best record.

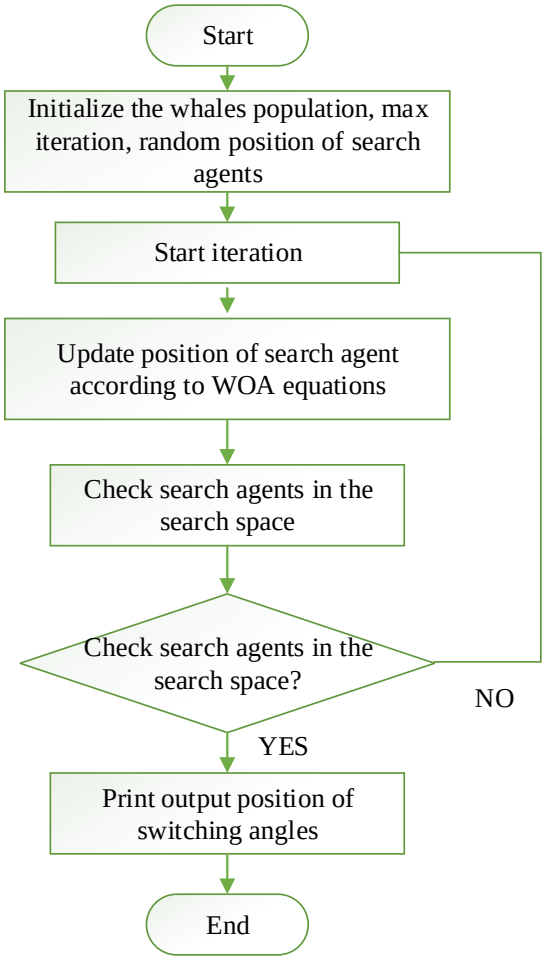


Figure 3.7: General Flowchart of WOA Algorithm.

The adaptive variation of the search vector *A* enables the WOA algorithm to transition seamlessly between exploration and exploitation: by reducing A, some iterations are devoted to exploration ($|A|>1$). At the same time, the remainder is committed to exploitation ($|A|<1$). Surprisingly, WOA requires the adjustment of only two primary internal factors (*A* and *C*). While mutation and other evolutionary operations might have been incorporated in the WOA formulation to mimic the behavior of humpback whales fully, we chose to minimize the number of heuristics and internal parameters, resulting in an actual implementation of the WOA algorithm. Hybridization with evolutionary search techniques, on the other hand, maybe the focus of future research.

3.5 Harris Hawks Optimization (HHO)

The Harris's hawk is unique because it cooperates with other family members living in the same stable group. In contrast, other raptors typically attack alone to discover and capture a quarry. This avian desert predator has evolved novel team chasing abilities that include tracing, encircling, flushing out, and ultimately attacking the potential quarry. During the non-breeding season, these clever birds can organize dinner parties for several individuals. In the raptor world, they are regarded as genuinely cooperative predators. They are familiar with their family members and attempt to anticipate their movements during the attack. When the party begins, and the hawks are assembled, they make brief tours and land on relatively high perches. In this manner, the hawks occasionally perform a "leapfrog" motion throughout the target area, rejoining and splitting several times to search for the covered animal actively, typically a rabbit [136].

Harris' hawks' primary method of prey capture is the "surprise pounce," also known as the "seven kills" strategy. In this intelligent strategy, multiple hawks attempt to attack cooperatively from different directions and simultaneously converge on a detected escaping rabbit outside the cover. While the attack can be completed quickly by capturing the surprising prey in a few seconds, depending on the prey's evasion capabilities and behaviors, the seven kills may occasionally include multiple, short-length, quick dives near the prey over several minutes. Harris' hawks can exhibit various chasing styles, depending on the dynamic nature of the situation and the prey's evasion patterns. When the best hawk (leader) stoops at the prey and becomes disoriented, the chase is continued by one of the party members. These switching behaviors can be observed in various contexts because they aid in confusing the escaping rabbit. The primary advantage of these cooperative tactics is that the Harris' hawks can pursue and exhaust the detected rabbit, increasing its vulnerability.

Additionally, by perplexing the escaping prey, it is unable to regain its defensive capabilities. Finally, it cannot flee the besieging team, as one of the hawks, who is

frequently the most powerful and experienced, effortlessly captures the exhausted rabbit and distributes it to other party members. Harris' hawks and their primary behaviors can be observed in the wild.

3.5.1 Phase of exploration

This section proposes a mechanism for HHO exploration. If we consider the nature of Harris' hawks, their powerful eyes enable them to track and detect prey, but the prey may not always be visible. Thus, the hawks wait, observe, and monitor the desert site in the hope of spotting prey several hours later. Harris' hawks are considered candidate solutions in HHO, and the best candidate solution at each step is considered to be the intended prey or nearly so. In HHO, Harris' hawks perch in random locations and wait for prey using two strategies. Suppose we assume an equal probability for each perching strategy. In that case, they either perch based on the positions of other family members (to be close enough to attack) and the rabbit, as modeled in Equation 3.22 for the condition of, or they perch on random tall trees (random locations within the group's home range), as modeled in Equation 3.22 for the condition of.

$$X_i(t+1) = \begin{cases} X_{rand}(t) - r_1 * |X_{rand}(t) - 2 * r_2 * X_i(t)|, & q \geq 0.5 \\ \left(X_{prey}(t) - X_m(t)\right) - r_3 * \left(lb + r_4 * (ub - lb)\right), & q < 0.5 \end{cases} \quad (3.22)$$

where $X_i(t+1)$ is the position vector of hawks in the next iteration, $X_{prey}(t)$ is the position of the rabbit, $X_m(t)$ is the current position vector of hawks, and $X_{rand}(t)$ are random numbers within (0,1) that are updated in each iteration. ub and lb indicate the upper and lower bounds of variables, is a randomly chosen hawk from the current population and is the average position of the current population of hawks.

We developed a straightforward technique for randomly generating places within the group's home range. The first rule creates solutions based on the position of a random

hawk and the presence of other hawks. In the second rule of Equation 3.22, we have the difference between the location of the best so far and the group's average position plus a randomly scaled component based on the range of variables, where is a scaling coefficient to increase the rule's random nature further once it approaches one and similar distribution patterns may occur. It is possible to design several updating algorithms, but we choose the most straightforward method to mimic hawk behavior. Equation 3.23 is used to get the average position of hawks:

$$X_m(t) = \frac{1}{M}\sum_{i=1}^{M} X_i(t) \tag{3.23}$$

where represents the iteration's location of each hawk and the total number of hawks. There are several methods for determining the average establishment, but we chose the easiest.

3.5.2 Exploration to exploitation transition

The HHO algorithm can transition from exploration to exploitation and then switch between several exploitative behaviors based on the prey's fleeing energy. A prey's energy level drops significantly during its evasion behavior. To account for this fact, the energy of a prey is calculated as follows:

$$E = 2 * E_0 * (1 - \frac{t}{t_{max}}) \tag{3.24}$$

where denotes the prey's fleeing energy, is the maximum number of rounds and denotes the prey's beginning level of energy. Each cycle of HHO causes a random change inside the interval (1, 1). When drops from 0 to 1, the rabbit is physically weakening, but when it increases from 0 to 1, the rabbit is strengthening. Throughout the iterations, the dynamic escape energy decreases. When the escaping energy is 1, the hawks seek various regions in quest of a rabbit position; hence, the HHO performs the exploration phase; and when the escaping energy is 1, the algorithm attempts to exploit the solutions' neighborhood during the exploitation stages. In

brief, exploration occurs during the first stage, whereas exploitation occurs throughout the subsequent steps.

3.5.3 Phase of exploitation

The Harris' hawks execute the surprise pounce in this phase by attacking the intended prey spotted in the preceding stage. Preys, on the other hand, frequently strive to flee dangerous situations. As a result, several pursuit styles arise in real-world circumstances. Four different tactics for modeling the attacking stage are proposed in the HHO based on prey escape behaviors and the chasing strategies of Harris' hawks.

Preys are constantly attempting to flee dangerous circumstances. Assume the probability of a prey successfully escaping or failing to escape (0.5) before a surprise pounce. Whatever the prey does, the hawks will engage in either a harsh or a soft besiege to capture it. This means that they will encircle the prey in various ways, softly or violently, depending on the amount of energy conserved by the prey. In real-world settings, the hawks approach the desired prey closer and closer to enhance their chances of collaboratively killing the rabbit by the surprise pounce. After a few minutes, the fleeing prey will begin to lose energy; at that point, the hawks will escalate their besiege technique to capture the fatigued victim effortlessly. The parameter is used to model this technique and enable the HHO to switch between soft and harsh besiege processes. In this regard, when 0.5 occurs, a gentle besiege occurs; when 0.5 occurs, a harsh besiege occurs.

3.5.3.1 Soft suffocation

When and, the rabbit retains sufficient energy to attempt to escape by a series of random deceptive hops, but it is ultimately unable to do so. The Harris' hawks encircle the rabbit softly during these attempts and then perform the surprise pounce. The following rules encapsulate this behavior:

$$X_i(t+1) = \triangle X_i(t) - E * |J * X_{prey}(t) - X_i(t)| \quad (3.25)$$

$$\triangle X_i(t) = X_{prey}(t) - X_i(t) \quad (3.26)$$

where is the gap between the rabbit's position vector and the iteration's current location, is a random number between (0,1), and is the rabbit's unexpected leap strength during the fleeing method. Each repetition, the value varies randomly to replicate the nature of rabbit motions.

3.5.3.2 Persistent encirclement

When the prey is between 0.5 and 0.5, it is exceptionally exhausted and has low fleeing energy. Furthermore, the Harris' hawks rarely orbit the intended target before eventually pounce. Equation 3.27 is used to update the current positions in this case:

$$X_i(t+1) = X_{prey}(t) - E * | \triangle X_i(t)| \quad (3.27)$$

3.5.3.3 Gradual fast dives with a soft besiege

When the rabbit is still 0.5, it has the energy to flee effectively, and a delicate besiege is prepared before the surprise pounce. This is a more clever procedure than the one described previously.

The HHO algorithm uses the levy flight (LF) idea to describe the prey's fleeing patterns and leapfrog motions mathematically. The LF is used to simulate the natural zigzag deceptive maneuvers of preys (namely rabbits) during the escape phase, as well as the irregular, sudden, and quick dives of hawks surrounding the escaping prey This mechanism is also corroborated by real-world observations of various forms of competition in nature. As a result, in this phase of the HHO approach, LF-based motions were used. Based on real-world hawk behavior, it is hypothesized that hawks can gradually select the optimum potential dive toward prey when competing

for prey. Thus, to conduct a mild besiege, we assume that the hawks can evaluate (decide) their next step following the following rule in Equation 3.28:

$$Y = X_{prey}(t) - E * |J * X_{rab} - X_i(t)| \tag{3.28}$$

Then, they compare the likely outcome of such a movement to the prior dive to determine whether it was a good dive. If this is not reasonable (when they notice the prey making more deceptive movements), they also begin to dive irregularly, abruptly, and rapidly when approaching the rabbit. We hypothesized that they would dive by the following rule:

$$Z = Y + S * levy(dim) \tag{3.29}$$

$$levy(x) = 0.01 \times \frac{u \times \sigma}{|\mu|^{\frac{1}{\beta}}}, \sigma = \left(\frac{\Gamma(1+\beta) \times \sin(\frac{\pi\beta}{2})}{\Gamma(\frac{\pi\beta}{2}) \times \beta \times 2^{(\frac{\beta-1}{2})}} \right)^{1/\beta} \tag{3.30}$$

where, $levy(x)$ denotes the Levy Flight model, with u and μ being random numbers in the range [0,1], and β, a constant value set to 1.5. S is a random vector set to 1 x dim, where *dim* is the number of preys. Y and Z define locations applied to all search agents. For an optimization problem of solving the minimum, the mathematical model is given as follows.

$$X_i(t+1) = \begin{cases} Y, & if \ f(Y) < f(X_i(t)) \\ Z, & if \ f(Z) < f(X_i(t)) \end{cases} \tag{3.31}$$

where f is the fitness value.

3.5.3.4 Hard besiege with progressive rapid dives

If $|E| < 0.5$ and $r > 0.5$, execute hard besiege with progressive rapid dives using Equations 3.32 and 3.33.

$$Y = X_{prey}(t) - E * |J * X_{prey}(t) - X_m(t)| \tag{3.32}$$

$$Z = Y + s * levy(dim) \tag{3.33}$$

In Figure 3.8, the flowchart of the HHO is repsented in detail.

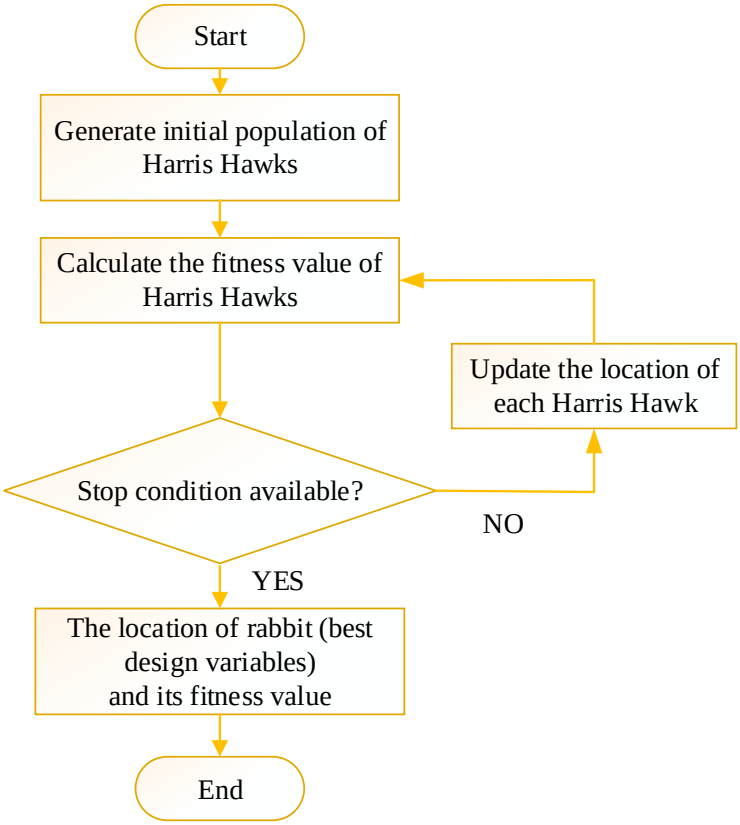


Figure 3.8: HHO Algorithm Flowchart

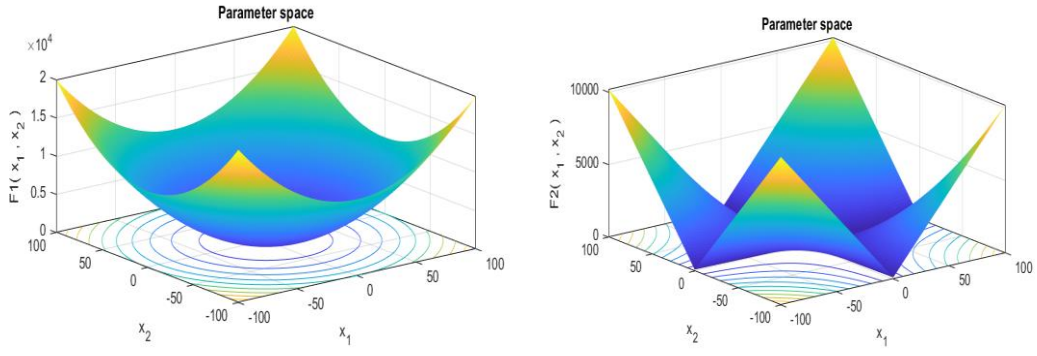
The cost function, the range of various optimization variables, and the determining optimal f_{min} quoted in the literature are summarized in Table 3.1. These benchmark functions represent shifted, rotated, enlarged, and combined variations of the most complicated mathematical optimization problems described in the literature. In Table 3.1, V_{no} denotes the number of model parameters. For each algorithm, a population size of 30 and a maximum iteration of 500 were used.

Table 3.1: Description of Unimodal Benchmark Functions

Function	V_{no}	**Range**	f_{min}
$F_1(x) = \sum_{i=1}^{n} x_i^2$	30	[-100,100]	0

$F_2(x) = \sum_{i=1}^{n} \lvert x_i \rvert + \prod_{i=1}^{n} \lvert x_i \rvert$	30	[-10,10]	0
$F_3(x) = \sum_{i=1}^{n} (\sum_{j-1}^{i} x_j)^2$	30	[-100,100]	0
$F_4(x) = \max\{\lvert x_i \rvert, 1 \leq i \leq n\}$	30	[-100,100]	0
$F_5(x) = \sum_{i=1}^{n} [100(x_{i+1} + x_i^2)^2 (x_i - 1)^2]$	30	[-30,30]	0
$F_6(x) = \sum_{i=1}^{n} (x_i + 0.5)^2$	30	[-100,100]	0

In Table 3.1. unimodal benchmark functions are specified in terms of their dimension V_{no}, upper and lower boundary ranges, and minimum value fmin. The F_1 function has a dimension of 30, has boundaries between -100 and 100, and a minimum value of 0. The F_2 function has a dimension of 30, has boundaries between -10 and 10, and a minimum value of 0. The F_3 function has a dimension of 30, has boundaries between -100 and 100, and a minimum value of 0. The F_4 function represented in Figure 3.14 (d) has a dimension of 30, has boundaries between -100 and 100, and a minimum value of 0. The F_5 function has a dimension of 30, has boundaries between -30 and 30, and a minimum value of 0. The F_6 function has a dimension of 30, has boundaries between -100 and 100, and a minimum value of 0.



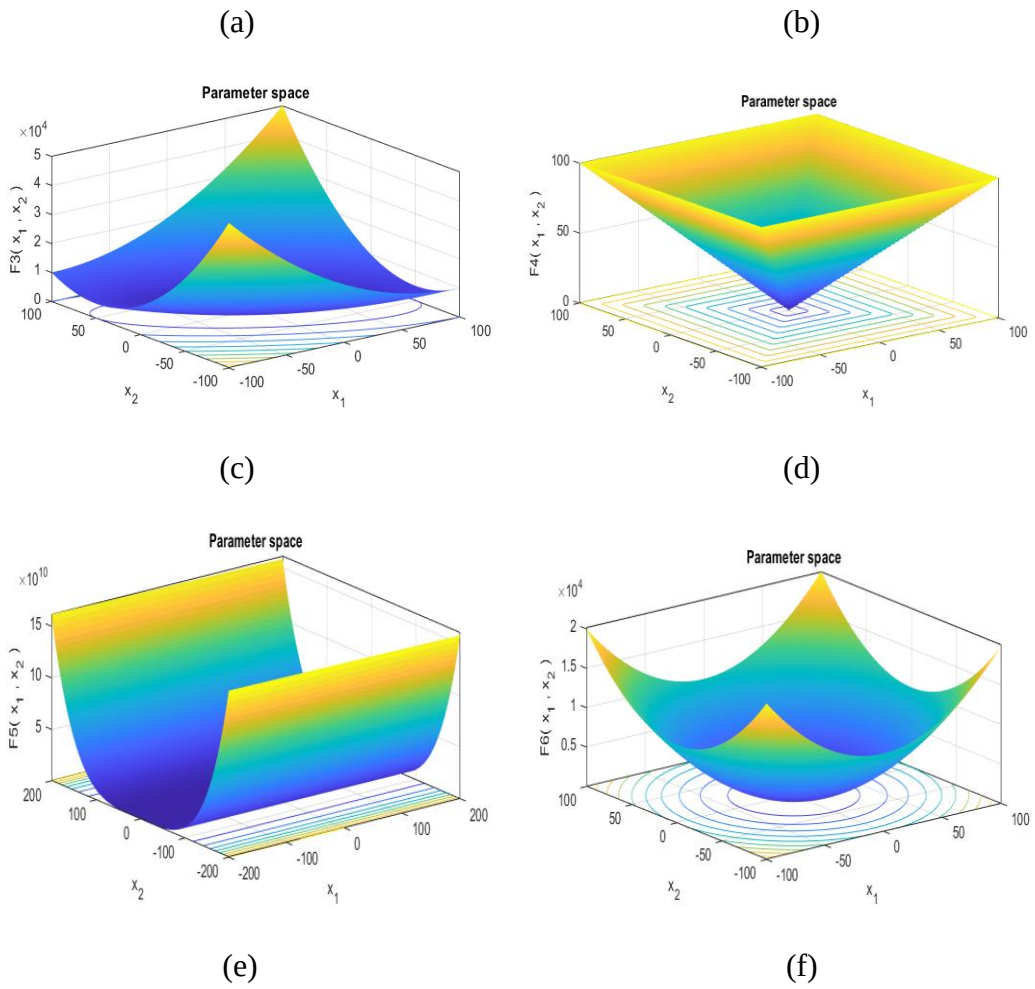


Figure 3.9: Benchmark Functions, (a) F_1 Function Parameter Space, (b) F_2 Function Parameter Space, (c) F_3 Function Parameter Space, (d) F_4 function Parameter Space, (e) F_5 Function Parameter Space, (f) F_6 Function Parameter Space

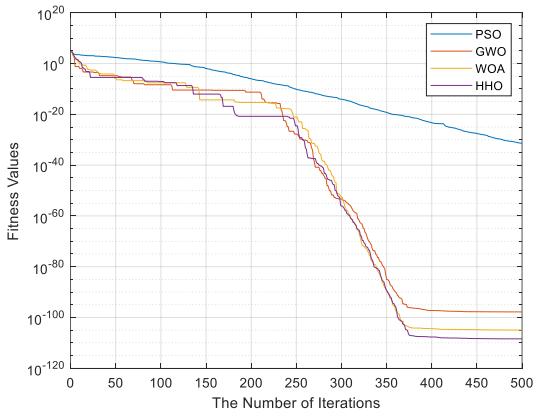


Figure 3.10: Convergence Curve of PSO, GWO, WOA, and HHO for F_1 Function

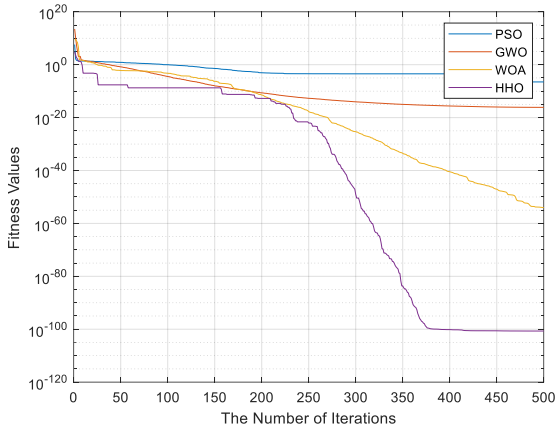


Figure 3.11: Convergence Curve of PSO, GWO, WOA, and HHO for F_2 Function

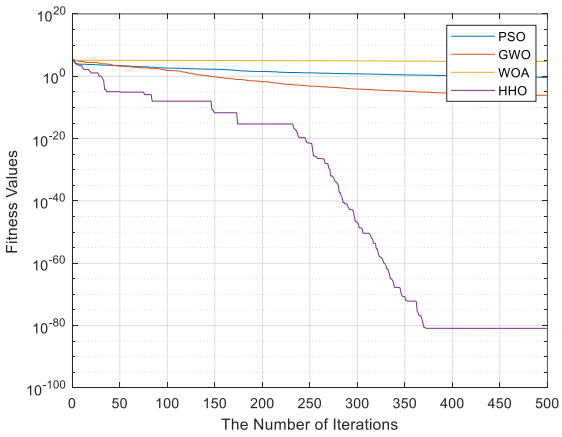


Figure 3.12: Convergence Curve of PSO, GWO, WOA, and HHO for F_3 Function

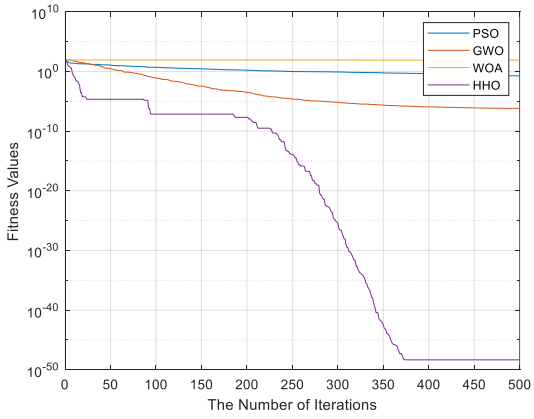


Figure 3.13: Convergence Curve of PSO, GWO, WOA, and HHO for F_4 Function

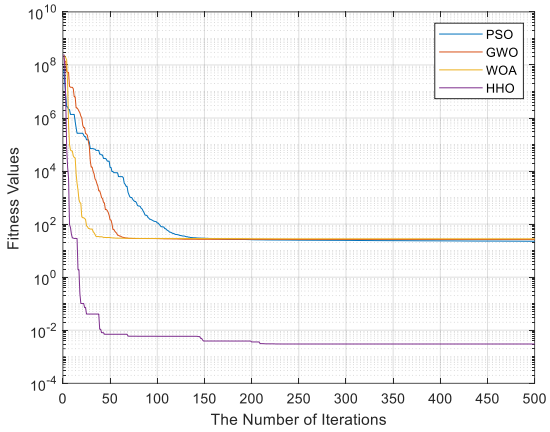


Figure 3.14: Convergence Curve of PSO, GWO, WOA, and HHO for F_5 Function

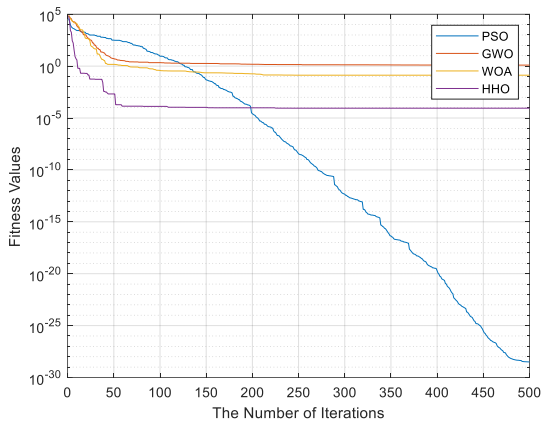


Figure 3.15: Convergence Curve of PSO, GWO, WOA, and HHO for F_6 Function

In figures 3.10-3.15, the fitness values of PSO, GWO, WOA, and HHO gradually decrease according to the number of iterations. The figures show that the HHO algorithm has superior performance than PSO, GWO, and WOA algorithms. HHO can be a better alternative than a higher convergence rate for electric drives, inverters, and SHE applications.

Pulse Width Modulation (PWM) development is used in inverters to give a reliable output voltage free of the heap. The inverters subject to the PWM development are superior to the standard inverters. Power switches are essential segments for PWM inverters. Selective harmonic elimination-pulse width modulation (SHE-PWM) employs an off-line switching angle calculation to eliminate undesired harmonic

content in the output AC voltage spectrum represented in Figure 3.3. To perform this, the switching angles are classically calculated for the first 90 of the output waveform, α_1, α_2, …, α_N as illustrated in Figure 3.16. Accordingly, symmetry is applied for the rest of the period to eliminate even harmonics in the resultant waveform.

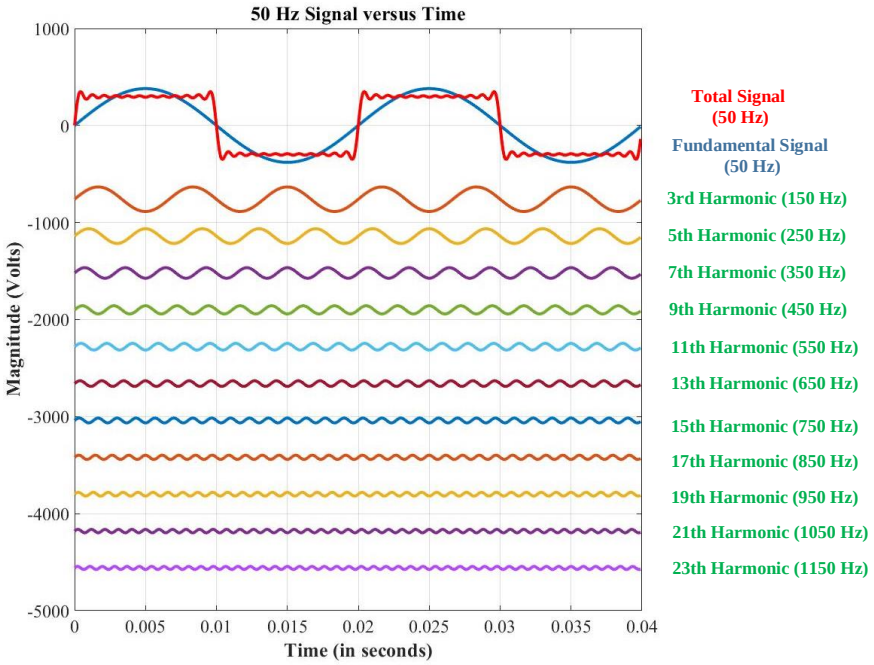


Figure 3.16: 50 Hz Voltage Signal and Harmonics up to 1150 Hz

These angle sets can normally be calculated off-line and then stored in a look-up table for later use, then in the online generation of the voltage waveform. Assuming that we want to obtain *k* calculated angles (α_1, α_2, …, α_k), a set of solutions can be achieved, equaling *k-1* harmonics to zero and providing a specific value to the amplitude of the fundamental component. Thus, at the output waveform, *k-1* harmonics are eliminated while the fundamental component results in the amplitude that we want to achieve. The switching frequency generated by this modulation type depends on the number of calculated angles and the frequency of the fundamental component of the output voltage. With these basics of SHE modulation, it is possible to obtain output voltage waveforms in which some undesired essential harmonics are

eliminated. This solution is useful in high power applications, such as railway traction, where a reduced switching frequency is desirable, and the harmonic content of the obtained waveform must be minimized.

The challenge in traction applications is to generate a three-phase alternating voltage output from a DC catenary voltage using the PWM approach. In SHEM, as many low order harmonics in the output AC voltage waveform as possible should be avoided, as these harmonics have a detrimental effect on the induction motor's performance. It is accomplished by designating a few pulses spread throughout a quarter of an entire AC voltage cycle. When the DC voltage is constant and the RMS value of the output voltage's fundamental component is constant, the starting and ending points of these voltage pulses in degrees or radians remain stable when the output frequency is adjusted. On the other hand, different RMS values of the output voltage at the same output frequency need distinct angles for the voltage pulses' start and finish points. If quarter-wave symmetry is maintained in SHEM, a preprogrammed modulation technique, there will be no even or triplen harmonics in the AC output voltage waveforms.

The output voltage waveforms retain only the fundamental component and odd harmonics. By defining optimal SHEM angles, one can theoretically eliminate N-1 odd harmonic components such as the 5th, 7th, 11th, 13th, and so on and estimate the magnitude of the fundamental component. SHEM is used to reduce low-order odd harmonics in line-to-line or line-to-neutral voltages referred to in the literature as TLL and TLN, respectively [127]. An odd number of SHEM angles is defined in this study work to detect (fix) the fundamental component and eliminate consecutive low order sidekick harmonics. In this case, N is an odd number, and the technique is referred to as the TLN1 technique. SHEM angles 1 to 7 are displayed in Figure 3.17 for N=7 in the first quarter of a line-to-neutral voltage cycle. These seven angles define the amplitude of the fundamental component and exclude N-1=6 low-order odd harmonics, i.e., the fifth, seventh, eleventh, thirteenth, thirteenth, seventeenth, and nineteenth harmonics.

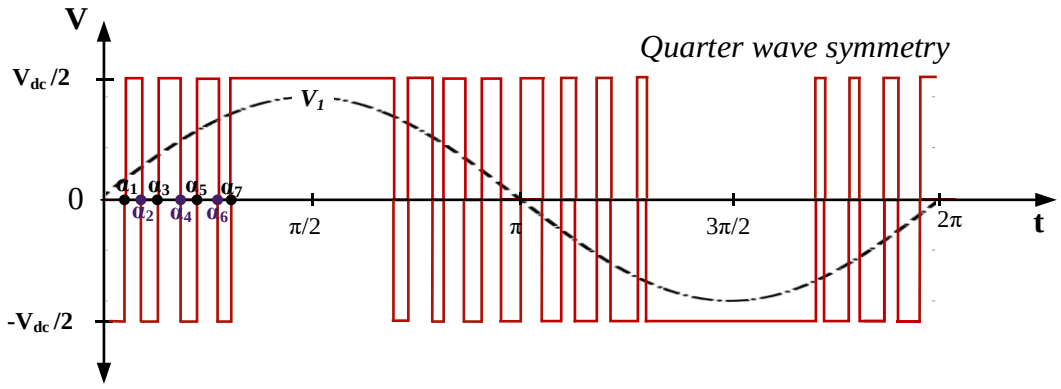


Figure 3.17: Definition of Seven SHEM Angles on Line-to-neutral Voltage Waveform Based on Quarter Wave Symmetry (Range of SHEM Angles in This Figure is 0-60°, $v_1(t)$ is the Fundamental Component of Voltage for Modulation Index M =0.8)

The Fourier series expansion of a distorted sine wave can be expressed as in Equation 3.34.

$$v(t) = a_0 + \sum_{n=1}^{\infty} a_n \cos(nw_0 t) + b_n \sin(nw_0 t) \tag{3.34}$$

where,

$$a_0 = \frac{1}{T}\int_{t_0}^{t_0+T} v(t)dt \tag{3.35}$$

$$a_n = \frac{2}{T}\int_{t_0}^{t_0+T} v(t)\cos(nw_0 t)dt \tag{3.36}$$

$$b_n = \frac{2}{T}\int_{t_0}^{t_0+T} v(t)sin(nw_0 t)dt \tag{3.37}$$

Since the inverter output voltage in Fig.3 has an odd quarter-wave symmetry, $a_0 = 0$, and $a_n = 0$. On the other hand, $b_n = 0$ for even harmonics. The expression given in Equation 3.34 can therefore be simplified, and corresponding b_n coefficients of the nth odd-order harmonic components can be derived from Equations 3.34 and 3.37, as expressed in Equation 3.38.

$$b_n = \frac{4}{n\pi}\frac{V_{dc}}{2}[\sum_{k=1}^{N}(-1)^k \cos(n\alpha_k)], \quad N\ is\ odd \tag{3.38}$$

where, b_n is the peak value of the n^{th} harmonic component in the line-to-neutral voltage waveform, with n = {5, 7, 11, … 50}, V_{dc}, the input DC link voltage, and α_k , the switching angles of SHEM, with k={1,2,…, N}.

Coefficient b_1 (for n=1), however, is the peak value of the fundamental line-to-neutral voltage component of the inverter, $\widehat{V_f}$ as expressed in Equation 3.39.

$$b_1 = \frac{4}{\pi}\frac{V_{dc}}{2}\left[\sum_{k=1}^{N}(-1)^k \cos(\alpha_k)\right] = \widehat{V_f} \tag{3.39}$$

The corresponding transcendental equation set in Equation 3.40 is then obtained, where N algebraic equations with N unknown α_k's (α_k, k=1,2, …N), are obtained. Here, the fundamental component is set to a predetermined output voltage value, and *(N-1)* selected harmonics to be eliminated are set to zero. Corresponding inverter switching frequency, f_{sw} can be calculated from $f_{sw} = (2N+1)f_1$ where N is the number of harmonics to be eliminated, and f_1 is the fundamental frequency.

$$H_1 = -1 - \left[2\sum_{k=1}^{N}(-1)^k \cos(\alpha_k)\right] - M = 0$$

$$H_5 = -1 - \left[2\sum_{k=1}^{N}(-1)^k \cos(5\alpha_k)\right] = 0$$

$$\vdots$$

$$H_n = -1 - \left[2\sum_{k=1}^{N}(-1)^k \cos(n\alpha_k)\right] = 0 \tag{3.40}$$

where the modulation index, $M = \frac{\pi\widehat{V_f}}{2V_{dc}}$.

The switching angle constraints are given in Equation 3.41:

$$0 < \alpha_1 < \alpha_2 < \alpha_3 < \cdots < \alpha_N < \frac{\pi}{2} \tag{3.41}$$

The problem's cost function, f_{cost} is given in Equation 3.42, which can be expressed by squaring and summing the left-hand sides of the equation sets given in Equation 3.40. The cost function, f_{cost} always takes positive values because of squaring terms.

$$f_{cost} = (\beta H_1^2 + H_5^2 + H_7^2 + \cdots + H_n^2) \quad (3.42)$$

where β is the penalty coefficient of the modulation index.

The minimum value of f_{cost} is equal to zero if the calculated switching angles from the optimization algorithm exactly satisfy the equation set given in (7). Metaheuristic algorithms such as PSO, GWO, WOA, and HHO have been proposed and compared in the literature to solve such sets of nonlinear algebraic equations. Harris Hawks Optimization (HHO) Algorithm is seen to be promising in this application as a bionic and intelligent algorithm suitable for problems with function optimization. In the next chapter, these algorithms solve the SHE problem, and the results are compared according to their THD performances.

CHAPTER 4

SIMULATION RESULTS OF METAHEURISTIC APPLICATIONS TO LIGHT-RAIL PUBLIC TRANSPORTATION SYSTEM

Light rail transit (LRT) is a mode of urban rail passenger transportation that combines tram and metro characteristics. While its rolling stock resembles a typical tram, it has a higher capacity and speed and frequently operates on an exclusive right-of-way. Light rail transit systems are more similar to and interchangeable with classic underground or at-grade subways and heavy rail metros in many places. TRAX, operated by the Utah Transit Authority, included 50 stations over three lines and was formerly one of the fastest-growing light rail systems in the country.

There is no universally accepted definition, but in the United States (where the term was coined in the 1970s from the engineering term light railway), light rail operates primarily along exclusive rights-of-way and employs either individual tramcars or multiple units coupled together to form a train with a lower capacity and speed than a long heavy-rail passenger train or metro system [137-142].

Several light rail networks exhibit more akin to rapid transit or even commuter rail; these heavier fast transit-like systems are sometimes referred to as light metros. Other weak rail networks resemble trams and operate partially on streets. Worldwide, light rail systems are located on all inhabited continents. They have grown in popularity over the last few years because of their lower capital costs and better reliability when compared to large rail systems.

Due to the different definitions of light rail, it is difficult to distinguish it from other urban and commuter rail forms. In one city, a system referred to as light rail may be termed a streetcar or tram system in another. On the other hand, several lines dubbed "light rail" are quite comparable to rapid transit; in recent years, new terminology such as light metro has been coined to characterize these medium-capacity systems.

Specific "light rail" systems, such as Sprinter, bore little resemblance to urban rail and might be classified as commuters or even intercity rail. In the United States, the phrase "light rail" refers to a broad range of passenger rail systems. The cost of these several forms of light rail travel varies significantly. Tram-like systems are frequently less expensive than metro-like systems.

The distinction between light rail and streetcar or tram systems is the most difficult to make. There is considerable overlap between the technologies; many of the identical vehicles may be used for both. Streetcars and trams are frequently classified as a subclass of light rail rather than a separate mode of transit. The two most common variants are as follows:

- The conventional kind, in which rails and trains run alongside streets and share space with automobile traffic. While stops are common, no effort is made to establish different stations. Due to the shared nature of the environment, the tracks are typically unnoticeable aesthetically.
- A more recent variant, trains typically operate on their right-of-way, distinct from road traffic. Stops generally are less in number, and vehicles are frequently boarded from a platform. Tracks are apparent, and considerable effort is made in some situations to keep traffic away by specific signaling, level crossings equipped with gate arms, or even complete separation via non-level junctions.

The Dulwich Hill Line in Sydney is primarily comprised of separated tracks that follow a historic heavy rail line. It can be difficult to discern a distinction between light rail and metros at the most significant degree of separation. Without the contrast between it and the rapid transit London Underground, the London Docklands Light Railway would almost certainly not be designated "light rail." In Europe and Asia, the term "light rail" is increasingly used to refer to any rapid transit system that operates at a lower frequency or with shorter trains than the London Underground or Singapore's Mass Rapid Transit. On closer examination, these technologies are more appropriately characterized as light metro or people movers. For example, Manila's

Line 1 and Line 3 are sometimes called "light rail," although they are entirely separated, primarily elevated rails. This phenomenon is widespread in East Asian cities, where elevated metro lines are called light rail lines in Shanghai, Wuhan, Dalian in China, and Jakarta, Greater Jakarta, and Palembang in Indonesia. In North America, these systems are not typically referred to as light rail.

The MTR Light Rail in Hong Kong provides service to the northwest suburbs with unidirectional high-floor LRVs. Numerous systems exhibit a mixture of traits. Indeed, with correct engineering, a rail line might be built along a street, then tunneled underground, and finally elevated viaduct. For example, the L Line "light rail" of the Los Angeles Metro Rail system has sections that could be classified as a tramway, a light metro, or, in a narrower sense, rapid transit. It is particularly prevalent in the United States, where there is little public awareness of the distinction between these many types of urban rail networks. The advancement of technology for low-floor and catenary-free trams enables the construction of such mixed systems with only brief and shallow underground sections beneath critical intersections. The clearance height required is significantly reduced compared to conventional light rail vehicles. It is also conceivable for high-floor rapid transit vehicles to run parallel to a street, much like a tram; this is referred to as road running.

Up to now, the light-rail transportation capacity reduction, increased capacity, multiple systems, and frequency of acceleration and braking are explained in detail. It is the purpose of this chapter to calculate SHE-PWM switching angle set tables utilizing PSO, GWO, WOA, and HHO algorithms under a variety of modulation index conditions, as previously stated. For each of these four approaches, the modulation index values are calculated at intervals of 0.1 units. In addition, the fitness values are provided in the tables to allow you to compare the results of different optimization algorithms. A light-rail system is formed of three parts, as seen in Figure 4.1. The first component is the input portion, followed by the inverter and the motor part. Gate signals generated by SHE-PWM in the control system block of the figure are used to regulate the operation of the inverter. The output voltage of the inverter can be modified under V/f control principles based on the value of the

modulation index. The motor has a power rating of 125 kW, a voltage of 480 V, a current of 200 A, a frequency of 60 Hz, and a speed of 1800 rpm. Motor resistance and inductance are calculated to be 0.024 and 0.285 mH, respectively, based on short- and open-circuit measurements and motor inductance is considered to be 0.8 mH for adding cable indcutance. This motor has a low inductance to eliminate or minimize harmonics, similar to a filter for working at low frequencies. For instance, a photovoltaic system may contain an L filter with a 5-20 mH value at the output stage. Thus, the residual high order frequency SHE technique can be readily eliminated, and the overall THD is smaller than projected.

Simulink simulations presented in Figure 4.1 are validated at 565 V minimum voltage and 750 V nominal DC-Link voltage values. All simulation results at 60 Hz for PSO, GWO, WOA, and HHO algorithms are replicated. Additionally, simulations for HHO are performed at 30 Hz and 60 Hz for various operating points of the motor with varying torque levels. After calculating and tabulating the switching angle sets, the simulations were carried out. Finally, there will be a section devoted to comparisons of the results of various metaheuristic algorithm simulations.

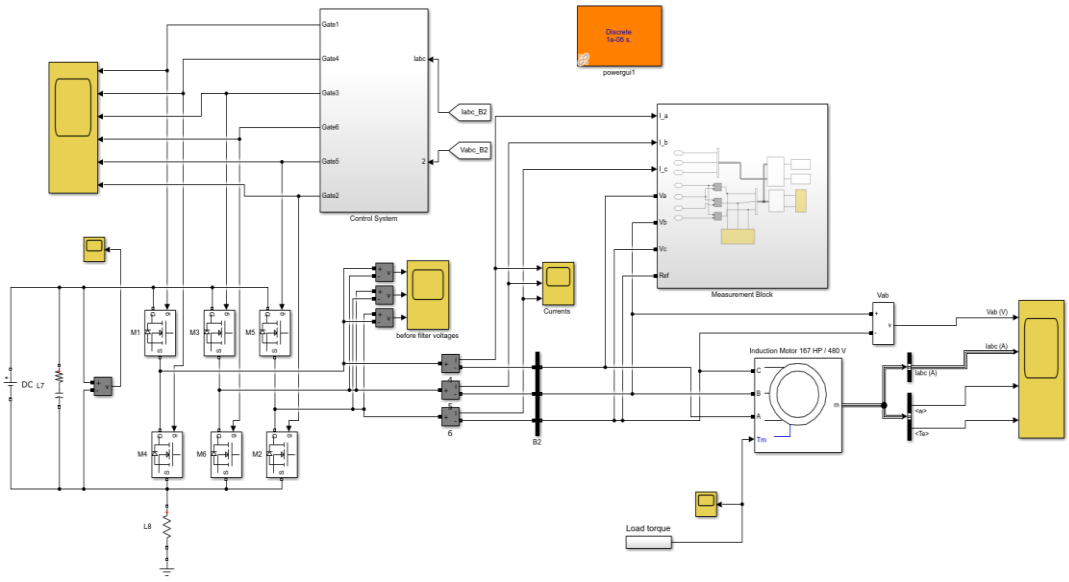


Figure 4.1: MATLAB/Simulink Circuit Diagram of LRT motor drive

4.1 PSO application to SHE-PWM

Particle swarm optimization (PSO) is a computational technique in computational science that aims to optimize a problem by iteratively attempting to improve a candidate solution on a given quality metric. It solves problems by populating the search space with candidate solutions, dubbed particles, and moving these particles around according to a basic mathematical formula over the particle's velocity and position. Its local best-known position guides each particle and the best-known positions in the search space, revised as other particles discover better positions. It should direct the swarm towards the optimal solutions. PSO algorithm is applied to SHE problem and calculated to switching angles to the light-rail transportation system. In these calculations, 500 iterations are made to obtain the needed fitness value and take minimum execution time. In Table 4.2, the switching angles are presented according to modulation index values.

Table 4.1: Switching Angles Varying with Modulation Index Values Calculated Using PSO Algorithm

Modulation index	**Switching angles (Degree)**							***f***
m	α_1	α_2	α_3	α_4	α_5	α_6	α_7	
0.10	14.01	15.62	29.67	30.31	44.72	45.46	59.72	$5.2x10^{-2}$
0.20	13.99	15.86	28.29	31.26	43.28	16.3	58.51	$1.8x10^{-2}$
0.30	12.70	16.49	27.04	31.04	42.56	47.32	57.54	$2.9x10^{-4}$
0.40	11.78	16.88	26.07	32.73	41.70	47.79	56.52	$4.1x10^{-3}$
0.50	10.07	16.85	25.83	32.49	40.01	48.67	55.61	$8.8x10^{-3}$
0.60	9.69	16.94	24.38	33.08	39.57	49.51	54.46	$1.0x10^{-3}$
0.70	8.33	16.36	23.52	33.73	38.38	49.70	53.45	$9.0x10^{-3}$
0.80	7.14	16.91	21.57	33.82	36.65	50.55	52.89	$1.0x10^{-3}$
0.90	6.45	15.31	19.40	31.23	33.45	48.15	48.69	$8.9x10^{-2}$
0.95	5.99	15.75	19.51	30.52	30.37	48.25	48.65	$2.4x10^{-2}$

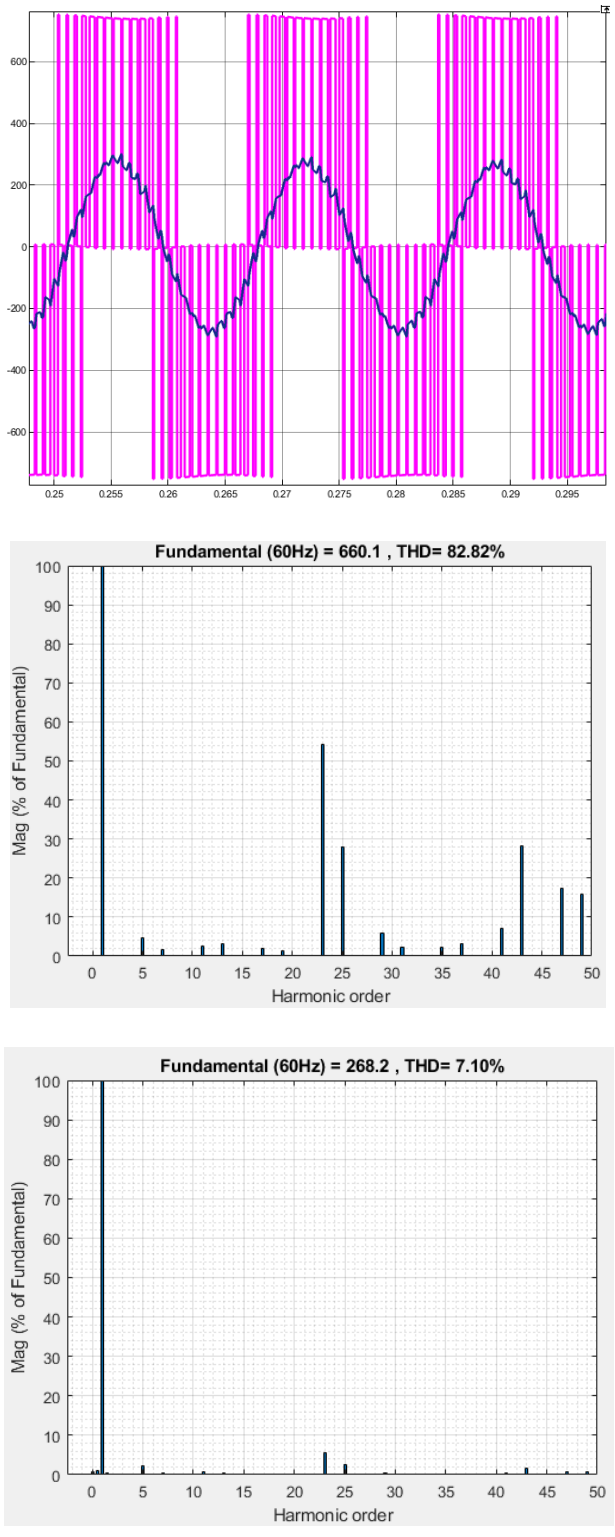


Figure 4.2: Inverter Output Voltage and Current Waveforms Obtained Switching Angles by PSO and Their FFT Distributions at 60 Hz, Full-Load, 750 V DC-Link Voltage

The switching angle sets specified in Table 4.2 with a modulation index of 0.8 are applied to the light-rail system depicted in Figure 4.1 in the Simulink environment. These implementation results are shown in Figure 4.2, together with their voltage, current, and FFT waveforms. DC-Link voltage is 750 V in this picture, as indicated by the magenta-colored peak voltage in the first portion. The inverter's output voltage is 660.1 V, and its THD value is 82.82 percent, composed of fifth, seventh, eleventh, thirteenth, thirteenth, seventeenth, and nineteenth harmonics. The line current is 268.2 A, and the THD value is 7.1 percent, as the motor inductance eliminates all other harmonics except for the fifth harmonic.

4.2 GWO application to SHE-PWM

Seyedali Mirjalili and colleagues proposed the grey wolf optimizer (GWO) in 2014 as a unique swarm intelligent optimization algorithm that primarily mimics the wolf leadership hierarchy and hunting process in nature. Seyedali and Mirjalili, among others, demonstrated that standard GWO outperforms PSO, GSA, DE, and FEP algorithms in terms of optimization performance [143]. Due to the wolf algorithm's inherent simplicity, rapid search speed, high search accuracy, and ease of implementation are more easily integrated with practical engineering challenges. As a result, GWO has considerable theoretical research value. However, because GWO is a novel biological intelligence algorithm, research and development of the theory are still in their infancy. Further study and research are required to improve the algorithm's performance.

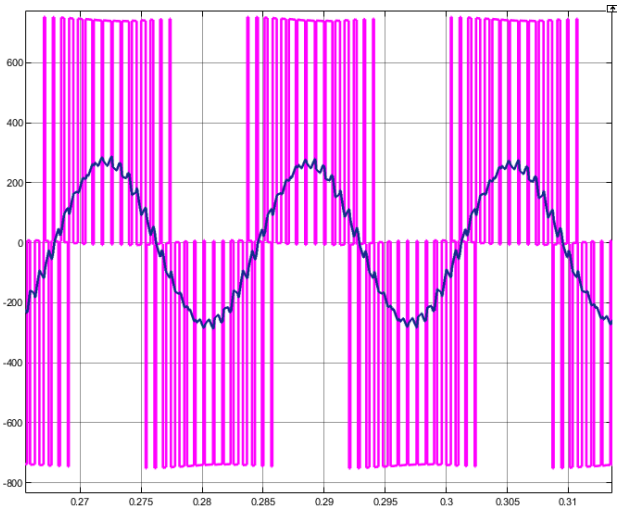
Numerous swarm-based algorithms are designed to replicate the hunting and searching activities of some animals. However, because GWO models the internal leadership structure of wolves, the position of the best answer can be evaluated entirely by three solutions during the search process. However, in other swarm intelligence, the optimal solution is found solely through the leadership of a single answer. Thus, GWO can significantly reduce the likelihood of being early and falling into the locally optimal.

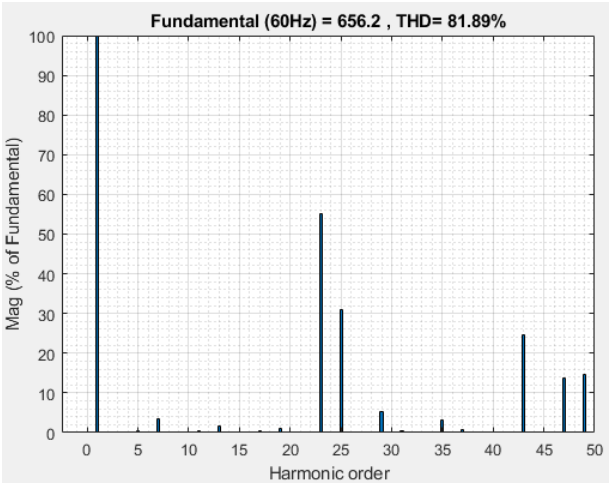
Table 4.2: Switching Angles Varying with Modulation Index Values Calculated Using GWO Algorithm

Modulation index	**Switching angles (Degree)**							**Fitness**
m	α_1	α_2	α_3	α_4	α_5	α_6	α_7	*f*
0.10	14.14	15.68	29.02	30.55	44.29	45.51	59.04	$8.6x10^{-3}$
0.20	13.42	15.44	28.08	31.37	43.82	46.88	58.03	$9.3x10^{-3}$
0.30	12.25	16.31	27.98	31.04	42.43	47.98	57.66	$1.8x10^{-3}$
0.40	11.17	16.82	26.62	32.69	41.98	47.72	56.12	$5.8x10^{-4}$
0.50	10.92	16.02	25.58	32.83	40.54	48.41	55.2	$4.2x10^{-4}$
0.60	9.69	16.93	24.07	33.66	39.33	49.05	54.97	$6.4x10^{-4}$
0.70	8.64	16.55	23.24	33.89	38.79	49.86	53.46	$2.0x10^{-5}$
0.80	7.96	16.40	21.49	33.69	36.34	50.31	52.69	$5.1x10^{-5}$
0.90	6.4	15.94	19.86	31.46	33.76	48.55	48.96	$8.7x10^{-4}$
0.95	6.25	14.59	17.24	29.89	30.57	40.07	48.11	$1.0x10^{-6}$

The switching angle sets described in Table 4.3 with a modulation index of 0.8 are applied to the Simulink environment's light-rail system represented in Figure 4.1. These implementation results, along with associated voltage, current, and FFT waveforms, are displayed in Figure 4.3. In this illustration, the DC-Link voltage is 750 V, as indicated by the magenta-colored peak voltage in the first section. The inverter's output voltage is 656.2 V, and the THD is 81.89 percent, composed of seventh, eleventh, and nineteenth harmonics dominated. Line current is 267.7 A, and THD is 6.88 percent, as the motor inductance cancels all other harmonics except the seventh harmonic.

When PSO and GWO are compared, these conclusions can be drawn: calculated switching angles by PSO are obtained a low THD signal but desired to eliminate harmonics magnitudes are higher than GWO. GWO is more successful at this point for eliminating all low-order harmonics, excluding the seventh harmonic.





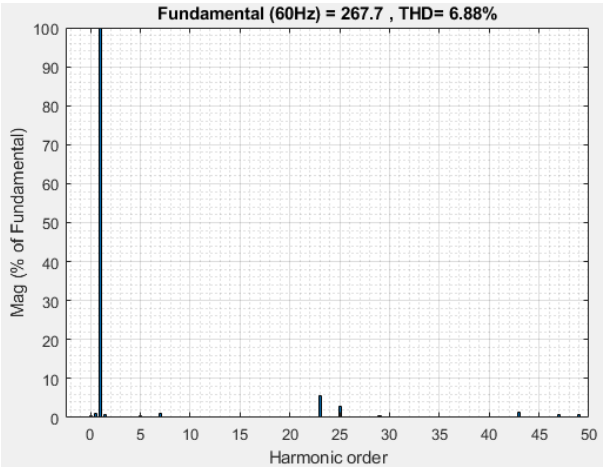


Figure 4.3: Inverter Output Voltage and Current Waveforms Obtained Switching Angles by GWO and Their FFT Distributions at 60 Hz, Full-Load, 750 V DC-Link Voltage

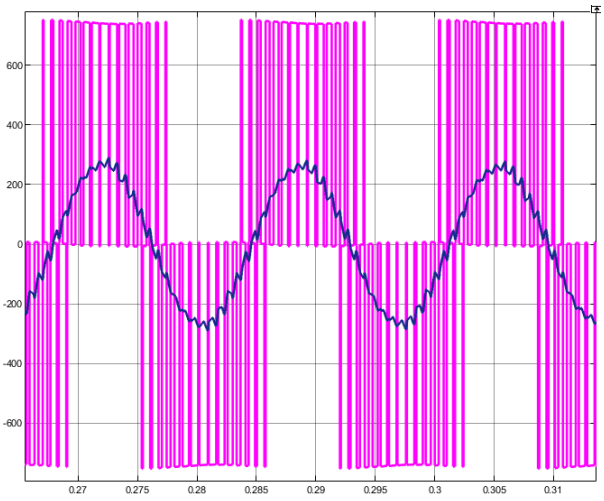
4.3 WOA application to SHE-PWM

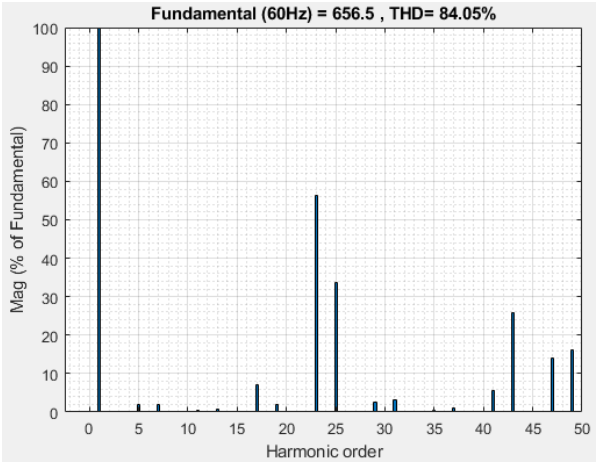
The whale optimization technique is a novel meta-heuristic optimization method that mimics the intelligence of humpback whales' bubble-net hunting behavior. WOA is a simple, robust, stochastic optimization approach based on swarm intelligence. Population-based WOA is capable of avoiding local optimum solutions in favor of a global optimum. Due to these characteristics, WOA is a suitable method for addressing many restricted or unconstrained optimization issues in practice without requiring structural reformation of the algorithm.

In the scope of WOA, a swarm is a collection of possible solutions to an optimization problem, with each possible solution denoted by a search agent. The WOA's objective is to determine the search agent location that optimizes a given fitness function.

Table 4.3: Switching Angles Varying with Modulation Index Values Calculated Using WOA

Modulation index	**Switching angles (Degree)**							**Fitness**
m	α_1	α_2	α_3	α_4	α_5	α_6	α_7	
0.10	14.18	15.69	29.14	30.95	44.01	45.12	59.43	$8.9\text{x}10^{-3}$
0.20	13.21	15.11	28.41	31.73	43.36	46.28	58.14	$1.1\text{x}10^{-3}$
0.30	12.56	16.49	27.26	31.34	42.48	47.9	57.99	$7.0\text{x}10^{-3}$
0.40	11.93	16.51	26.55	33.00	41.79	47.72	56.58	$2.4\text{x}10^{-4}$
0.50	10.6	16.07	25.49	32.08	40.27	48.94	55.32	$8.7\text{x}10^{-4}$
0.60	9.43	16.39	24.68	33.15	39.94	49.44	54.30	$3.1\text{x}10^{-4}$
0.70	8.14	16.20	23.41	33.85	38.72	49.80	53.36	$1.5\text{x}10^{-4}$
0.80	7.15	16.29	21.76	33.43	36.95	50.61	52.3	$5.3\text{x}10^{-4}$
0.90	6.82	15.45	19.56	31.98	33.56	48.23	48.55	$4.5\text{x}10^{-3}$
0.95	5.31	15.19	17.64	29.62	30.33	48.15	48.31	$9.1\text{x}10^{-3}$





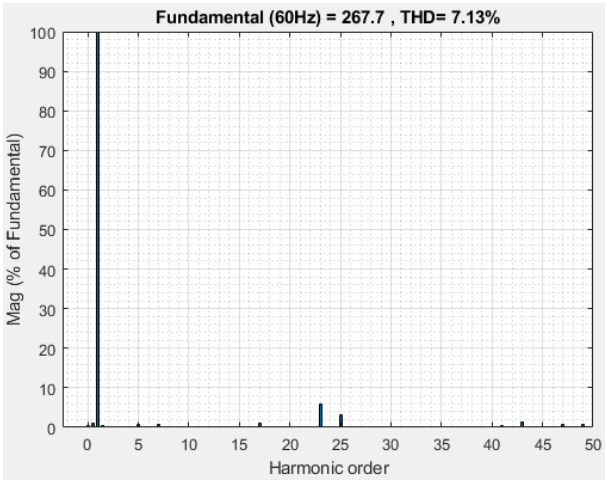


Figure 4.4: Inverter Output Voltage and Current Waveforms Obtained Switching Angles by WOA and Their FFT Distributions at 60 Hz, Full-Load, 750 V DC-Link Voltage

For the light-rail environment system represented in Figure 4.1, the angle switching sets described in Table 4.4 with modulation index 0.8 apply. Figure 4.4 shows these implementation results and the associated waveforms for voltage, current, and FFT. The DC-Link voltage is set to 750 V in this photo, as described in the first section as a magenta-colored peak voltage. The inverter output voltage is 656.5 V, and the THD is 84.05% consisting of 5th, 7th, 17th, and 19th harmonics. Line current is 267.7 A, and THD is 7.13%. The SHE switching angles are calculated for WOA. However, all harmonics cannot be removed by calculation sensitivity.

4.4 HHO application to SHE-PWM

Heidari et al. developed the Harris hawks optimizer (HHO), a swarm-based optimization approach as the mentioned previous chapter. The primary goal of HHO is to replicate Hawk's team collaboration hunting in nature and prey escaping to identify answers to the single-objective challenge. Hawks pursuing prey symbolize the search agent in HHO, while prey represents the optimal position.

HHO has a vital role to play in resolving a variety of real-world optimization challenges. Additionally, the HHO may be used to solve issues involving unknown types of search space and problems involving discrete and continuous areas, improve solution quality, extract optimal parameters with high accuracy, and improve prediction performance. Additionally, it established that HHO is a potentially powerful optimizer that aids in the solution of complex nonlinear problems by locating the best solution quickly and using a straightforward computational approach.

The switching angle sets given in Table 4.3 with a modulation index of 0.8 are applied to the light-rail system depicted in Figure 4.1 in the Simulink environment. Figure 4.3 illustrates these implementation findings, together with the accompanying voltage, current, and FFT waveforms.

Table 4.4: Switching Angles Varying with Modulation Index Values Calculated Using HHO

Modulation index	**Switching angles (Degree)**							**Fitness**
m	α_1	α_2	α_3	α_4	α_5	α_6	α_7	f
0.10	14.16	15.28	29.19	30.52	44.21	45.69	59.23	$1.1x10^{-4}$
0.20	13.46	15.63	28.39	31.06	43.37	46.35	58.45	$2.3x10^{-4}$
0.30	12.37	16.05	27.23	31.76	42.21	47.27	57.37	$1.5x10^{-4}$
0.40	11.51	16.35	26.29	32.28	41.26	47.99	55.52	$1.4x10^{-4}$
0.50	10.81	16.56	25.52	32.66	40.48	48.55	55.81	$1.1x10^{-5}$
0.60	9.75	16.8	24.31	33.15	39.26	49.33	54.73	$2.3x10^{-6}$
0.70	8.92	16.89	23.31	33.39	38.21	49.87	53.85	$7.2x10^{-6}$
0.80	7.78	16.76	21.8	33.32	36.48	50.33	52.46	$6.2x10^{-6}$
0.90	6.52	15.81	19.63	31.46	33.25	47.63	48.14	$2.6x10^{-8}$
0.95	6.11	14.99	18.36	29.72	31.02	45.8	46.05	$9.7x10^{-4}$

In Figure 4.5, the DC-Link voltage is 565 V at full-load, as indicated by the magenta-colored peak voltage in the first portion. The inverter's output voltage is 503 V, and the THD is 80.38 percent, eliminating all desired harmonics. The line current is 227.4 A, and the total harmonic distortion is 5.9 percent, as the motor inductance cancels all other harmonics. The findings of voltage and current signals and their FFT graphs at 60 Hz and half-load are shown in Figure 4.6. In Figure 4.6, the DC-Link voltage is 565 V. Current THD is greater than that of full-load waveforms. The identical voltage waveforms at 30 Hz are shown in Figure 4.7, but with increased THD values. There are no low-order harmonics in these FFT waveforms. All experiments in Figure 4.8-4.10 are conducted at 750 V DC-Link voltage and output frequencies of 60 and 30 Hz. As illustrated in Figure 4.8, the THD distribution of voltage and current waveforms at 60 Hz and full load does not contain the fifth, seventh, eleventh, thirteenth, thirteenth, seventeenth, and nineteenth harmonics. Similarly, the voltage and current signals in Figures 4.9 and 4.10 at 60 Hz half-load and 30 Hz full-load do not include low-order harmonics.

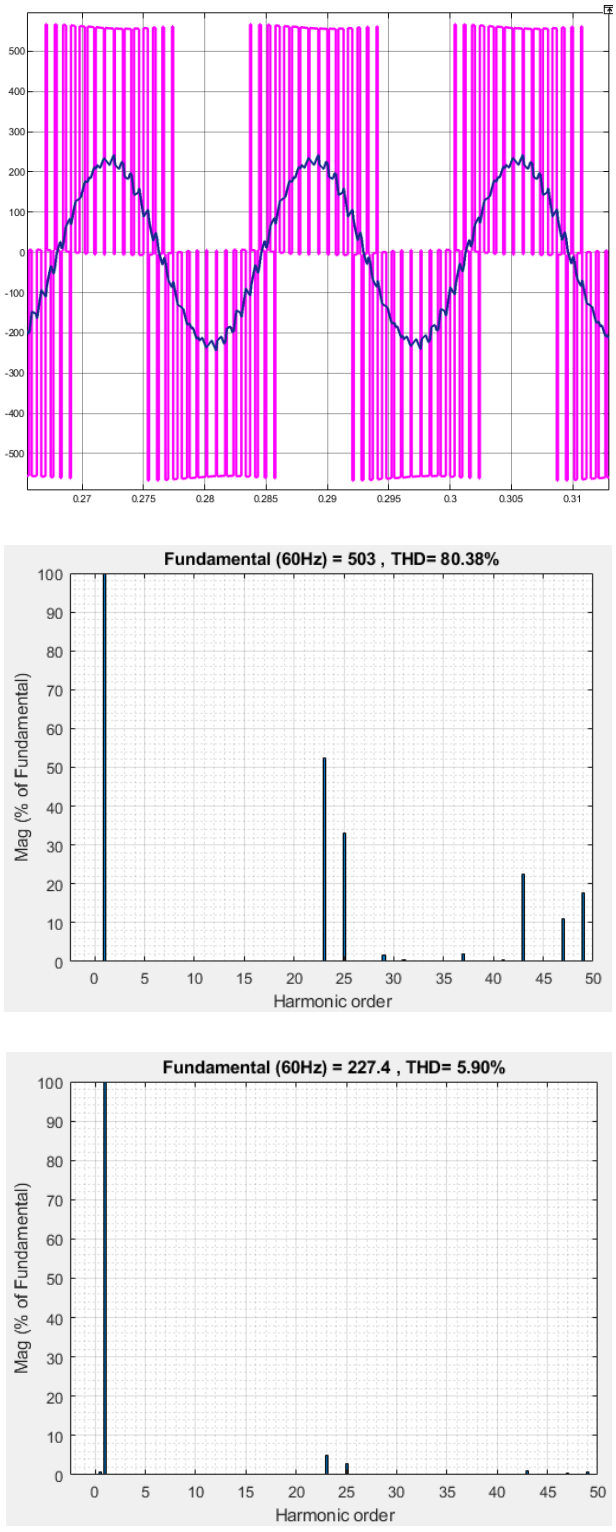


Figure 4.5: Inverter Output Voltage and Current Waveforms Obtained Switching Angles by HHO and Their FFT Distributions at 60 Hz, Full-Load, 565 V DC-Link Voltage

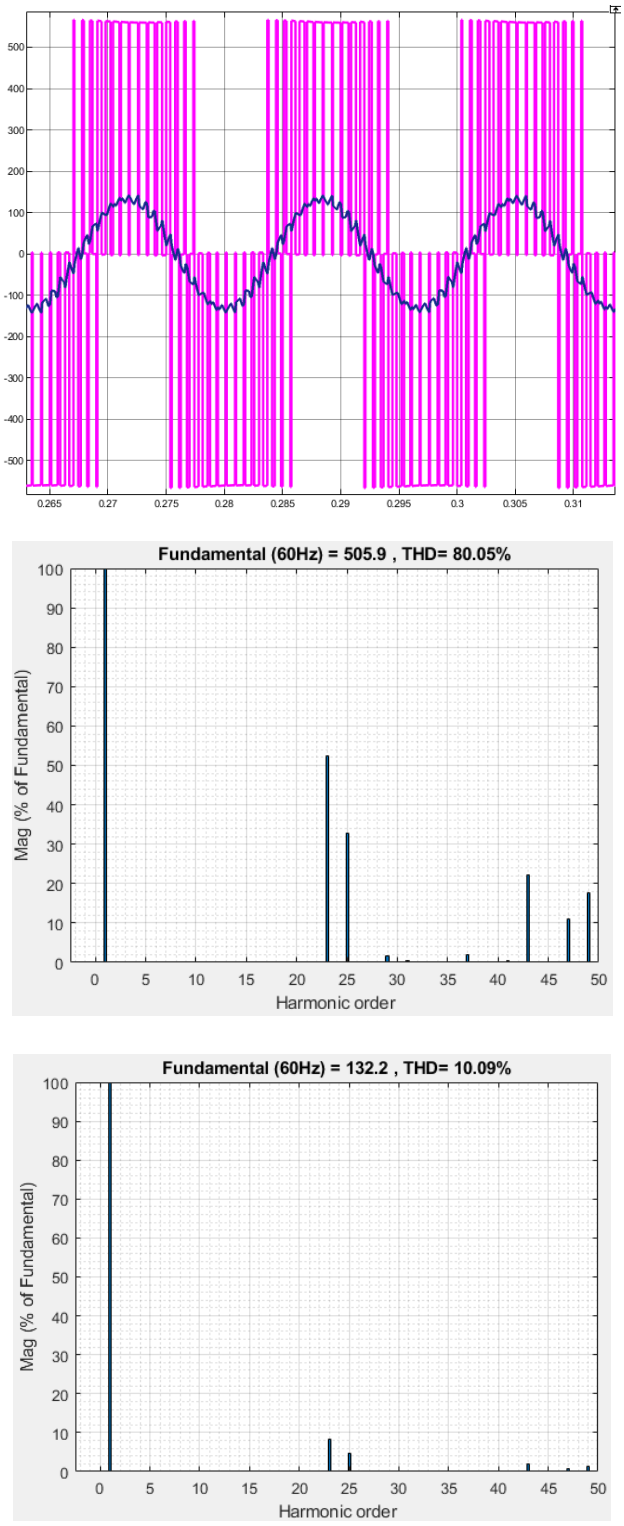


Figure 4.6: Inverter Output Voltage and Current Waveforms Obtained Switching Angles by HHO and Their FFT Distributions at 60 Hz, Half-Load, 565 V DC-Link Voltage

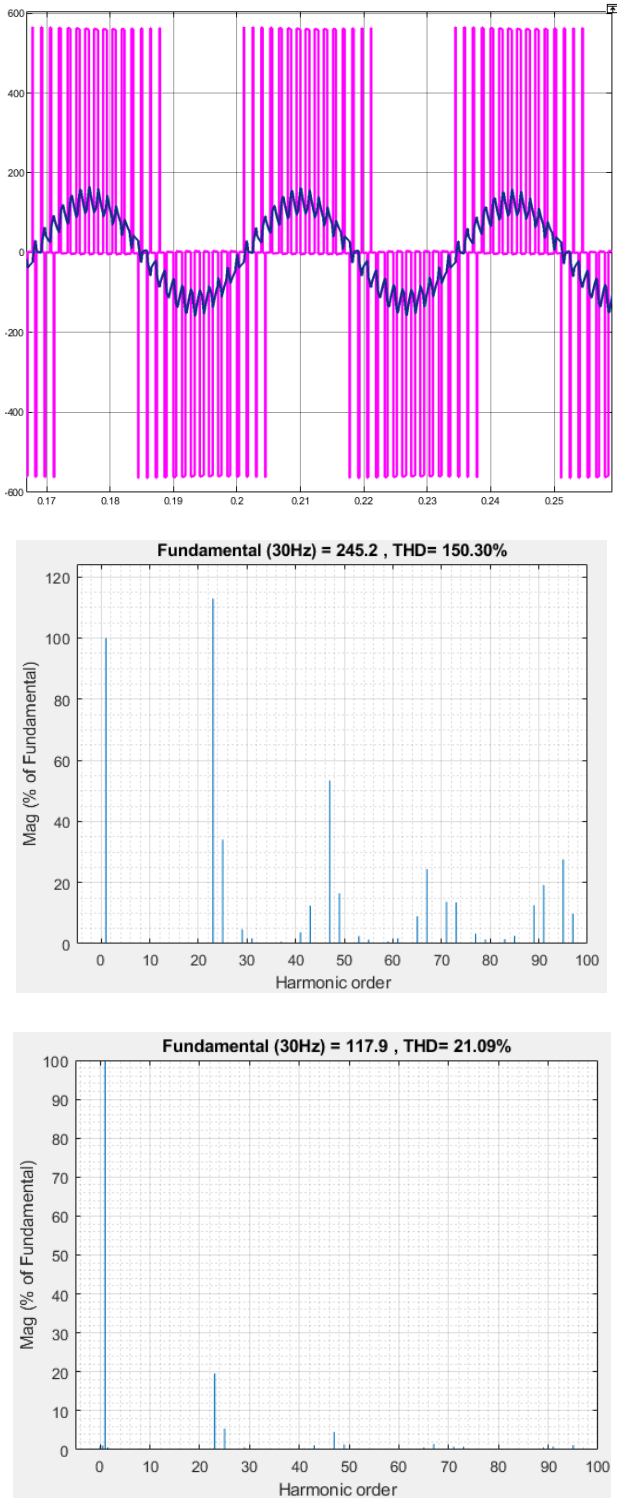


Figure 4.7: Inverter Output Voltage and Current Waveforms Obtained Switching Angles by HHO and Their FFT Distributions at 30 Hz, Full-Load, 565 V DC-Link Voltage

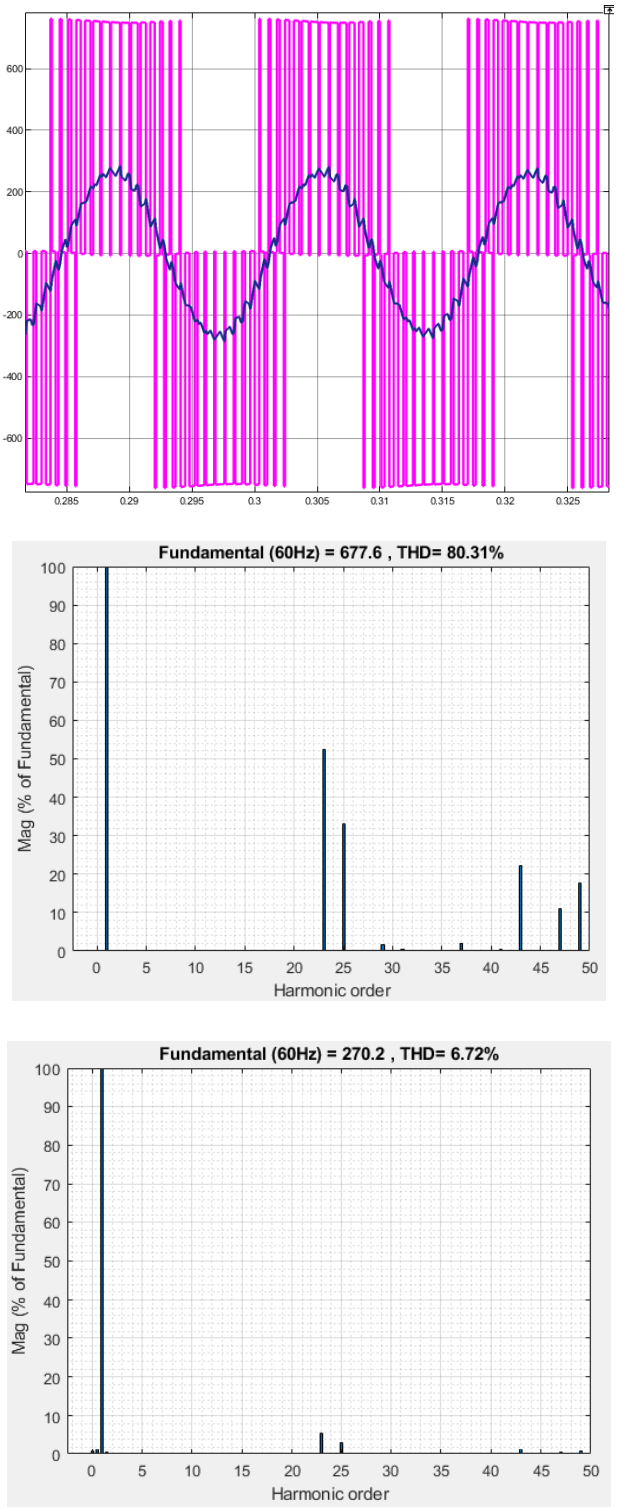


Figure 4.8: Inverter Output Voltage and Current Waveforms Obtained Switching Angles by HHO and Their FFT Distributions at 60 Hz, Full-Load, 750 V DC-Link Voltage

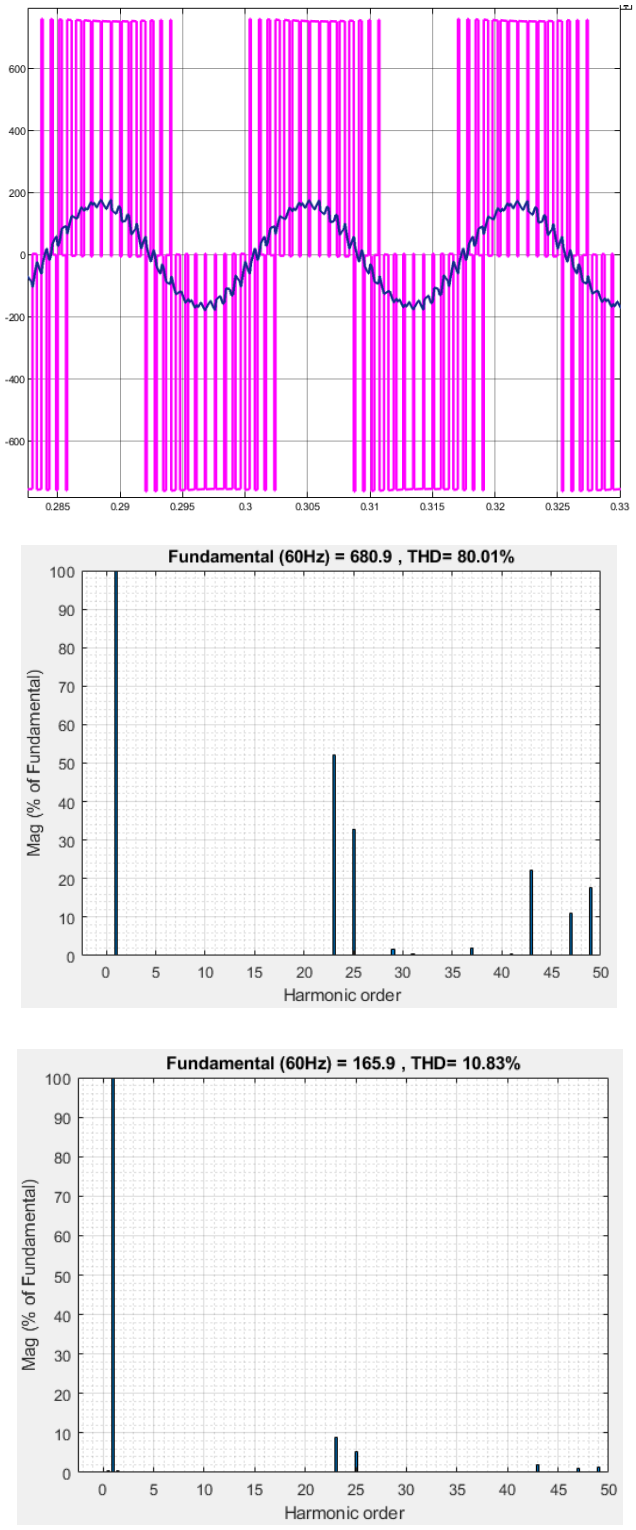
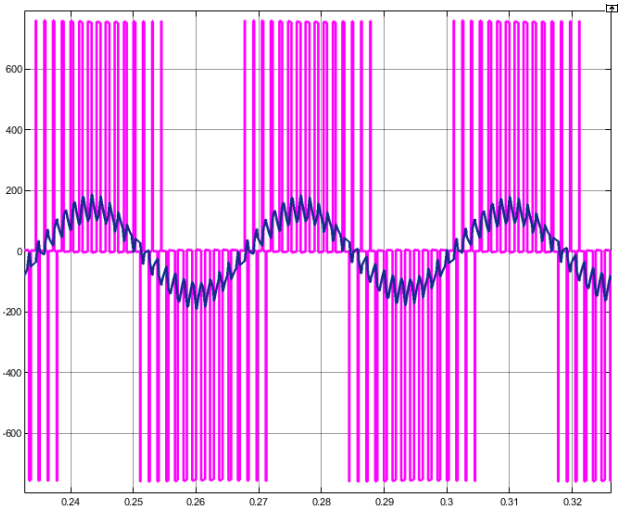
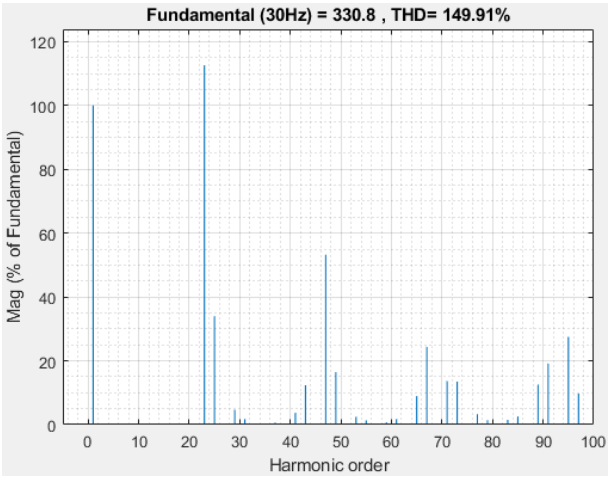


Figure 4.9: Inverter Output Voltage and Current Waveforms Obtained Switching Angles by HHO and Their FFT Distributions at 60 Hz, Half-Load, 750 V DC-Link Voltage





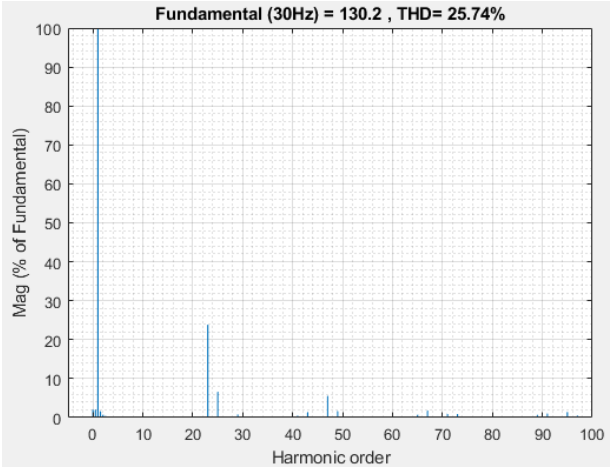


Figure 4.10: Inverter Output Voltage and Current Waveforms Obtained Switching Angles by HHO and Their FFT Distributions at 30 Hz, Full-Load, 750 V DC-Link Voltage

CHAPTER 5

COMPARISON OF RESULTS OBTAINED FROM TITAN AND JETSON TX1

The graphics processing unit (GPU) is a critical component of modern computing systems. It is a highly parallel, programmable, and powerful single-chip processor designed for computing. The GPU represented in Figure 5.1 was created to meet high-performance computing requirements, parallelism, and latency over throughput.

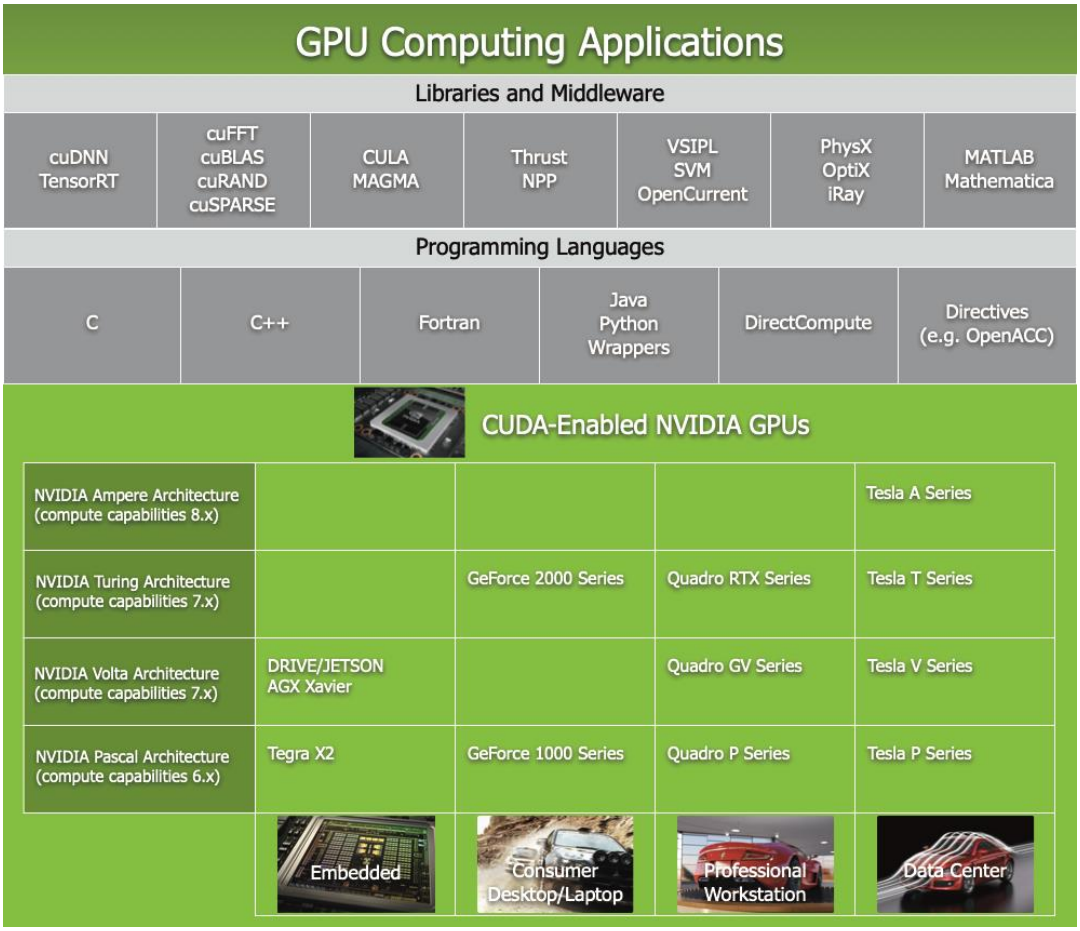


Figure 5.1: GPU computing applications

5.1 Evolution of Graphics Processing Unit Computing

In 1999, NVIDIA (a multinational technology company headquartered in Santa Clara, California) introduced the world's first GPU, the "GeForce 256." It is a single-chip real-time three-dimensional (3D) graphics processor. It featured 32 MB (Megabytes) of 128-bit DRAM, 23 million transistors, a 120MHz (Megahertz) core clock, and a 64-bit rendering pipeline. GPUs were initially used to perform floating-point arithmetic on three-dimensional geometry and vertices and then to manage pixel lighting and color values in high-definition graphics. The evolution of modern GPUs began with programmability from fixed-function pipelines to microcode processors, programmable processors, configurable processors, and scalable parallel processors.

The evolution of GPU architecture began with a single-core fixed-function hardware pipeline implementation for graphics. It progressed to a collection of fully programmable and highly parallel cores for general-purpose computation.

NVIDIA released "GeForce 3" in 2001 and continued to release faster, more precise, and programmable GPUs such as Geforce FX, ATI Radeon 9700, and ATI Radeon 8500. Simultaneously, Microsoft released a series of GPU computing drivers, and Intel began releasing new generations of its graphics processing units. By 2003, with the introduction of DirectX 9, GPU hardware included programmable GPUs. These cards supported both floating-point and advanced texture processing units. Additionally, in 2004, the GPU programming language appeared in software that supported real-time conditionals, loops, and dynamic flow controls. The GPU architecture was created to achieve high precision, speed, multiple rendering buffers, increased GPU memory, and texture access. These fixed-function graphics pipeline GPUs had processors dedicated to specific functions such as vertex, triangle, pixel, ROP, and memory.

Although the GPU was originally designed solely for graphics, it has since evolved to include accuracy, computing, and performance. A graphics pipeline is composed of stages that involve the combination of abstract models via a graphics processing

unit (GPU) core (hardware) and central processing unit (CPU) package (software). It has aided in accelerating the adoption of GPU computing technology by enabling high-speed computation. GPUs are integrated into the organization's current setups, ranging from desktop and laptop computers to mobile handsets and supercomputers.

5.2 GPU Computing Trends

The future of computation is computation, and the end of analysis is the GPU. There are three major GPU vendors in the PC market: Intel, AMD, and NVIDIA. Intel is the market leader in integrated and low-performance graphics, while AMD and NVIDIA dominate the high-performance discrete graphics market. NVIDIA is the market leader in both academic and industrial environments.

Today, combining CPU and GPU capabilities is a popular way to achieve the benefits of both CPU and GPU computing hardware and software solutions. This trend toward CPU-like GPU computing continues to be benefited by both emerging general-purpose technologies and existing general-purpose technologies. NVIDIA launched the first Fermi graphics processing unit (GPU), which was optimized for GPGPU (General-Purpose Graphics Processing Unit) computing. Fermi GPUs featured a unified memory address space, a proper hardware cache hierarchy, improved double-precision performance, concurrent kernel execution, and dual wrap schedulers.

Today, GPU computing is used in gaming, virtual reality (VR), artificial intelligence (AI), and high-performance computing (HPC), as well as self-driving cars and autonomous machines. Integrating computer graphics, GPU computing, and artificial intelligence is advancing our world by enabling future technologies such as autonomous vehicles, smart cities, and drug discovery.

The thesis studies use granted Titan V by NVIDIA and Jetson TX1 graphic units to calculate SHE is switching angles for gate drivers of three-phase two-level VSIs. These GPUs will be explained and compared in detail in this chapter.

5.2.1 Titan V

NVIDIA's TITAN V graphics card shown in Figure 5.2, is an enthusiast-class graphics card released on December 7th, 2017. The card, built on the 12 nm technology and is based on the GV100 graphics processor in its GV100-400-A1 edition, supports DirectX 12 and is built on the GV100 graphics processor. It assures that all current video games will function appropriately on TITAN V. The GV100 graphics processor has a die area of 815 mm2 and a transistor count of 21,100 million, making it a big device. It has 5120 shading units, 320 texture mapping units, and 96 ROPs, among other things. As a bonus, the system comes with 640 tensor cores, which accelerate machine learning applications. To maximize performance, NVIDIA has combined the TITAN V with 12 GB of HBM2 memory, which is connected through a 3072-bit memory interface. The graphics processing unit (GPU) operates at a frequency of 1200 MHz, which can be increased to 1455 MHz if necessary, and the memory operates at 848 MHz.

Figure 5.2: NVIDIA TITAN V graphic card

- Six hundred forty tensor cores are available, and performance has increased by order of magnitude. Every business requires artificial intelligence, and because of this tremendous increase in speed, AI can now be deployed across

all industries. NVIDIA Volta is a deep learning processor with 640 Tensor Cores that produces over 100 Teraflops per second (TFLOPS) of deep learning capability, representing a 5X increase in performance over the previous generation NVIDIA Pascal architecture.

- New graphic processing unit (GPU) architecture engineered to Work with Today's Computer The most difficult tasks facing humanity will necessitate the most powerful computing engine possible for computational and data research. In terms of transistor count, Volta is the most powerful GPU architecture the world has ever seen, with over 21 billion transistors. CUDA and Tensor Cores are combined to perform an AI supercomputer in a graphics processing unit (GPU).
- Scalability of the nest generation NVLink for faster time-to-solution Volta makes advantage of the innovative NVIDIA NVLink high-speed connection technology, which is the next generation. When compared to the previous generation of NVLink, this gives two times the throughput. It opens the door to more complex model and data-parallel techniques for large-scale scaling, allowing the maximum possible application performance.
- Volta-optimized software GPU-accelerated frameworks and applications presented by data scientists are frequently compelled to make trade-offs between model accuracy and lengthier run-times to achieve optimal results. With Volta-optimized CUDA and NVIDIA Deep Learning SDK libraries such as cuDNN, NCCL, and TensorRT, the industry's most popular frameworks and applications can easily make use of Volta's capabilities and performance. This accelerates the pace at which data scientists and academics make discoveries compared to previous years.

5.2.2 Jetson TX1

The NVIDIA Jetson TX1 presented in Figure 5.3 is a system-on-module (SoM) solution for visual computing applications. It is based on the NVIDIA Jetson

architecture. The newest NVIDIA Maxwell GPU architecture and a quad-core ARM Cortex-A57 MPCore (Quad-Core) CPU cluster can deliver the performance and power efficiency demanded by industry-leading visual computing applications for next-generation devices.

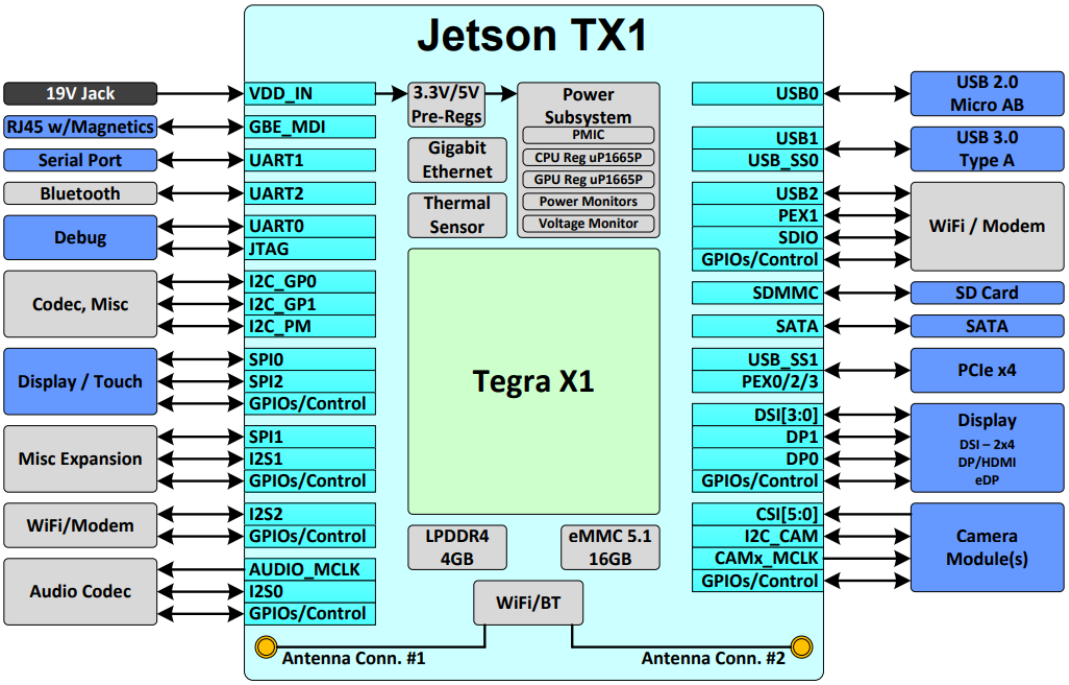


Figure 5.3: Jetson TX1 Block Diagram for general application

The Jetson TX1 SoM, which is intended for usage in power-constrained environments, includes the following features: Advanced 3D graphics, video, and

image processing; Parallel computing, computer vision, and machine learning capabilities;

- Operating system supports for both 32-bit and 64-bit architectures.
- The combination of remarkable performance and power economy, integrated capabilities, extensive I/O, and tiny size enables new product classes while simultaneously reducing the complexity of system integration and system integration costs.
- In addition to these uses, the Jetson TX1 is also well-suited for a variety of others, including:
    - ✓ Intelligent video analytics (IVA),
    - ✓ Telepresence drones,
    - ✓ Robotic systems,
    - ✓ Gaming devices,
    - ✓ Augmented reality, and
    - ✓ Portable medical devices.

5.3 CUDA Programming

Compute unified device architecture (CUDA) programming allows us to use NVIDIA's parallel computing features. The CUDA architecture and application programming interface (API) are especially advantageous for implementing general-purpose computation on graphics processing units (GPU). Although the interface is written in C/C++, it supports additional computer languages and frameworks. The CUDA platform provides complete access to GPU instruction sets and parallel computational units. Although CUDA's interface is written in C/C++, you can use any programming language or framework, including OpenCL and HIP.

The CUDA programming model enables you to write a scalar program while leveraging parallel programming for GPU resource allocation. The CUDA compiler makes use of programming abstractions to make use of built-in parallelism. CUDA

programmers can leverage three essential language extensions: CUDA blocks, shared memory, and synchronization barriers. CUDA blocks are composed of threads.

Multiple threads can share memory and halt until all threads reach a specified set of execution. The following code illustrates how the CUDA kernel adds vectors A and B and returns their product, vector C. The algorithm is designed to process scalars due to the execution of two vectors. This code can be used to simplify enormous parallelism. When running on a GPU, each vector element is executed by a thread, and the CUDA block's threads run independently and in parallel.

```
/** CUDA kernel device code - CUDA Sample Codes
 * Computes the vector addition of A and B into C. The
three vectors have the same number of elements as
implements.
 */
__global__ void vectorAdd( float *A, float *B, float
*C, int numElements) {
  int i = blockDim.x * blockIdx.x + threadIdx.x;
  if (i < numElements) {
    C[i] = A[i] + B[i];
  }
}
```

The CUDA programming model enables software to be scaled by dynamically expanding the number of GPU processing cores. You can program applications using CUDA language abstractions, dividing your programs into separate discrete tasks.

Further decomposing minor problems into smaller chunks of code is possible without disrupting processes, as parallel threads within the CUDA block continue to execute and cooperate. The CUDA runtime schedules and orders the execution of CUDA blocks on multiprocessors, allowing the CUDA program to run on any number of multiprocessors.

This procedure is illustrated in Figure 5.4, which depicts a CUDA program composed of eight CUDA blocks. The figure shows how the CUDA runtime selects

which blocks to assign to streaming multiprocessors (SMs). The tiny GPU with four SMs uses two CUDA blocks per SM, whereas the larger GPU with eight SMs uses one CUDA block per SM. This allocation type enables you to ensure performance scalability without having to adjust the code.

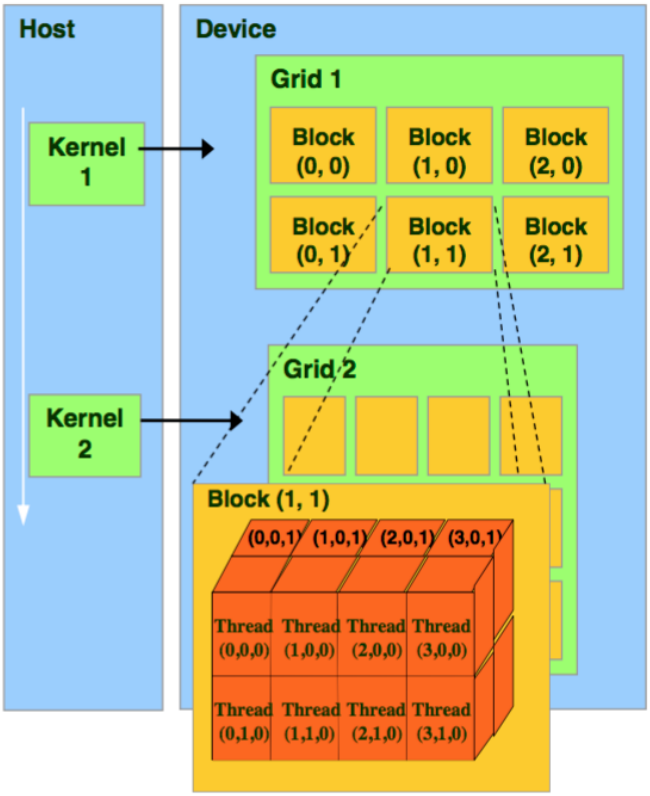


Figure 5.4: CUDA block scalability across different kinds of GPUs

CUDA programs often contain code instructions for both the GPU and the CPU, and the default C program includes a CUDA program and the host code. CPUs are referred to as hosts in this topology, while GPUs are referred to as devices. Each requires a distinct compiler. What you can do is as follows:

- You can compile the host code using a standard C compiler such as GCC.
- To enable devices to understand API functions, specific compilers are required. For NVIDIA GPUs, you can utilize the NVIDIA C Compiler (NVCC).

- The NVCC compiler is capable of decoupling the host and device code in a CUDA program. It is accomplished by invoking particular CUDA commands. The device code is annotated with keywords that refer to data-parallel functions, collectively referred to as 'Kernels.' After identifying these keywords, the NVCC compiles and runs the device code on the GPU.

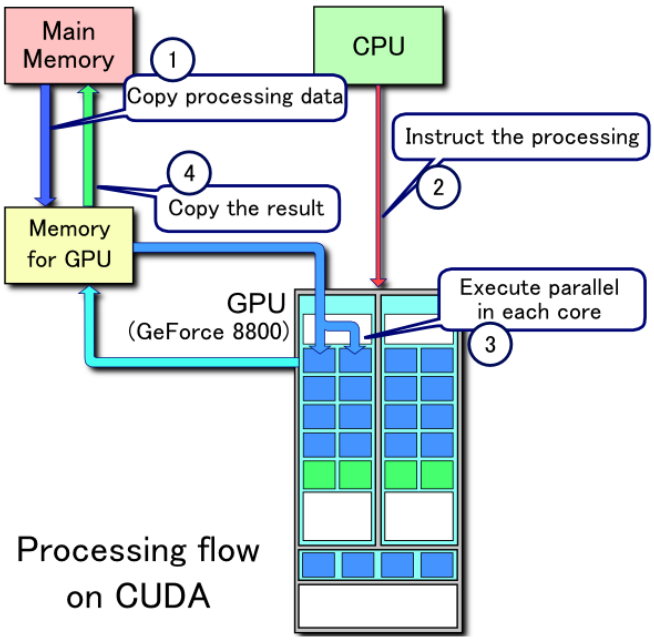


Figure 5.5: The flowchart of GPU programming

When you write a CUDA application, you have complete control over the number of launched threads. However, you should proceed cautiously. Threads are strung together to form blocks, which are then strung together to form three-dimensional grids. Each thread is assigned a unique identifier, which controls the execution of data. Typically, each GPU includes integrated global memory, referred to as dynamic random access memory (DRAM) or device memory. To run a kernel on a GPU, you must build code that allocates dedicated GPU memory. It is accomplished by utilizing the CUDA API's particular functionalities. The following diagram illustrates how this process works:

- Configure the device's memory allocation.
- Transfer data from the host computer's memory to the device's memory.

- On the device, execute the kernel.
- Return the result to the host memory from the device memory.
- Release the device's allotted memory.

The host can then access the device's memory and transfer data to and from it. However, the device is incapable of data transfer to and from the host. The CUDA program structure is stored on two machines: the host computer that runs the program and the GPU that executes the CUDA code. Each storage process adheres to the C memory paradigm, with its memory stack and heap. It means that data must be transferred independently from the host to the device. Transfer in some circumstances entails programming code manually to copy memory from one area to another. However, if you're using NVIDIA, you may use unified memory, eliminating the need for custom coding and saving time. This paradigm permits memory allocation from CPUs and GPUs, as well as memory prefetching before usage.

5.4 Comparison of Titan V and Jetson TX1 Results

In real-time industrial applications, the control process time of electronics boards plays an important role. PSO, GWO, WOA, and HHO metaheuristic algorithm CUDA codes are given in Appendix A, B, C, and D, respectively. These codes are run at Titan V with Intel Core i9 Processor and Jetson TX1. For all two systems, the CUDA C++ programming language is preferred to speed up the processes.

Table 5.1: Comparison of Titan V and Jetson TX1 processing times of algorithms

Metaheuristic Method	Titan V time(µs)	Jetson TX1 time(µs)	MATLAB CPU(ms)
PSO	236	295	95.8
GWO	157.1	212.3	78.1
WOA	110.8	138.5	55.7
HHO	80.4	124.8	18.3

According to Table 5.1, all metaheuristic methods can be used for SHE-PWM real-time applications because of microseconds level calculations. But the calculated switching angles are not accurate like MATLAB software. For example, for modulation index 0.7, the voltage THD of switching angles calculated with Titan V is 81.38%, while the other estimated value with MATLAB software is 80.38%. This THD value obtained from Titan V can be improved with the help of professional code written in CUDA, but the processing time will increase. Compared to these methods, HHO has superior performance of fast calculations.

With a critical breakdown strength ten times greater than silicon, silicon carbide represented in Figure 6.1 MOSFETs can operate at substantially higher temperatures, deliver more current density, have lower switching losses, and support significantly higher switching frequencies. It also means that silicon carbide MOSFETs are more comparable to silicon IGBTs. In many designs, silicon carbide MOSFETs can be used in place of silicon IGBTs while providing extra advantages to the overall design.

Another way silicon carbide MOSFETs surpass their silicon counterparts is in their ability to manage increased voltage and power demands while still conserving space. Because of the usage of silicon carbide, these MOSFETs are exceptionally tough and long-lasting.

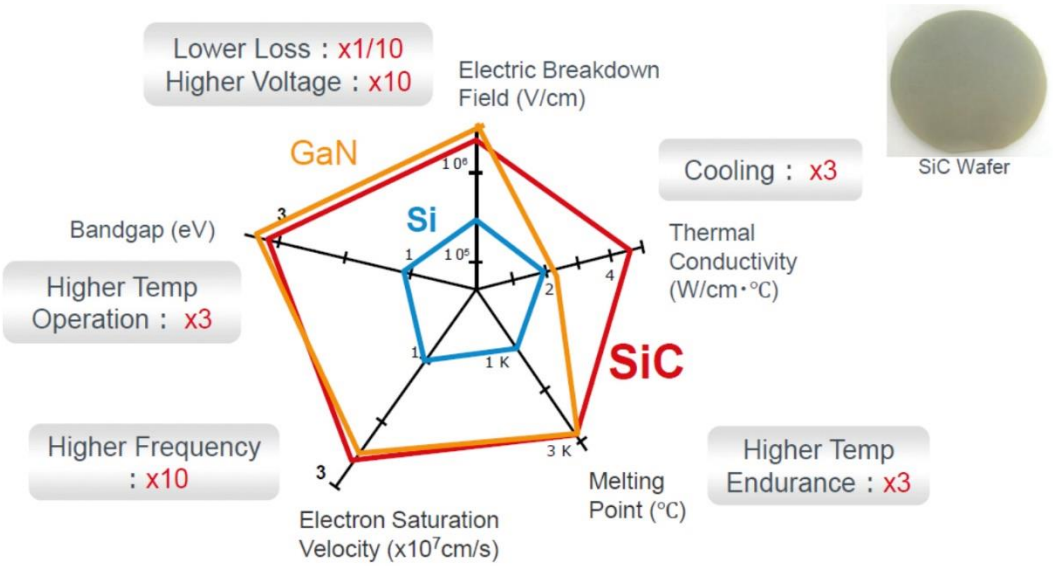


Figure 6.1: Semiconductor Material Comparison of Si, SiC, and GaN

IGBTs are employed in applications where there is a requirement for well-controlled, medium-speed switching, and they can be significantly less expensive than equivalent silicon MOSFETs in some cases. In addition, IGBTs can withstand higher voltages than standard MOSFETs, although this comes at the expense of significant

switching losses when silicon is utilized in the construction. Since these losses generate heat, it is necessary to implement costly and large thermal management solutions, which in turn reduces the efficiency of the power-conversion system.

Simply using a silicon IGBT will increase the size and weight of the system by a significant amount, which might be a severe concern for designs that involve electric vehicles or aircraft applications. IGBTs, on the other hand, provide excellent efficiency and energy savings at lower switching rates, which is why they have long been chosen over equivalent MOSFETs in many applications.

Because of the outstanding thermal conductivity of silicon carbide MOSFETs, they have higher thermal conductivity and lower switching losses than conventional MOSFETs. For the simple reason of lower switching losses (even at high voltages), systems utilizing silicon carbide MOSFETs as opposed to silicon IGBTs generate significantly less heat, resulting in a reduction in the thermal management needs of the system in question.

As a result, overall expenses are reduced, and the design is significantly more compact and lightweight when compared to previous models. Furthermore, silicon carbide MOSFETs are more durable than silicon IGBTs, making them excellent for use in severe environments where IGBTs would be ineffective, such as onboard chargers for electric vehicles or solar power generation systems.

To simulate a 750 V dc catenary line, a PWM rectifier is developed, a traction inverter is produced to drive the traction motor, and a bidirectional dc–dc converter is established to charge and discharge the supercapacitor bank. All traction systems operating on 750 V dc catenaries should be capable of withstanding 500–900 V continuously, 1000 V for 5 minutes, and 1000–1270 V in a straight line for durations ranging from 1 s to 20 ms, as specified in Annex A of EN 50163: 2004+A1:2007 [156]. That is the primary reason why all developed power converters use 1700 V/325 A at 25 °C SiC Power MOSFETS (Cree CAS300M17BM2) rather than 1200 V devices. All power converters that have been developed [53].

In a real system, a 750 V dc catenary line and its voltage variations are implemented by a PWM rectifier, which is achieved as a three-phase two-level SiC power MOSFET-based power electronic converter, which has a faster response time, lower current harmonic distortion, and increased performance when compared to IGBT-based alternatives. Figure 6.2 depicts the power circuit for the SiC PWM rectifier and the control circuitry for the rectifier. Due to the bidirectional power transfer capacity of the PWM rectifier, the regenerated power provided by the traction system during braking periods can be transferred to the alternating current grid in the event that the supercapacitor ESS is not in operation. This property of the PWM rectifiers allows them to be connected to the current transformer substations in parallel with the high-capacity bridge rectifier, which is impossible with other rectifiers.

A traction inverter represented in Figure 6.2 is designed to drive the traction motor. A PWM rectifier is shown in the figure expressed as a DC catenary simulator is established to model a 750 V DC catenary line. All traction systems for 750 V DC catenaries, as recommended, can withstand 500-900 V continuously, 1000 V for 5 minutes, and 1000-1270 V on a straight line for durations from 1 to 20 ms. That is the principal motivation why in all the established power converters, 1700 V / 325 A at 25ºC case temperature SiC Power MOSFETs (Cree CAS300M170BM2) are implemented instead of 1200 V units. The rectifier's switching frequency is kept constant at 10 kHz, but the inverter switching frequency is various at 750 Hz to 10 kHz in this article.

The traction inverter represented in Figure 6.2 built-in this article operates from a catenary of 750 V dc, whose operating voltage is allowed to vary between 500 and 900 V, respectively, as the lowest and the highest continuous voltage applied following EN50163:2014. Since the traction inverter will work for less than 5 minutes at non-permanent voltages between 900-1000 V DC, is the catenary voltage stays within this range or more than 1000 V, SiC power MOSFET signals are automatically blocked.

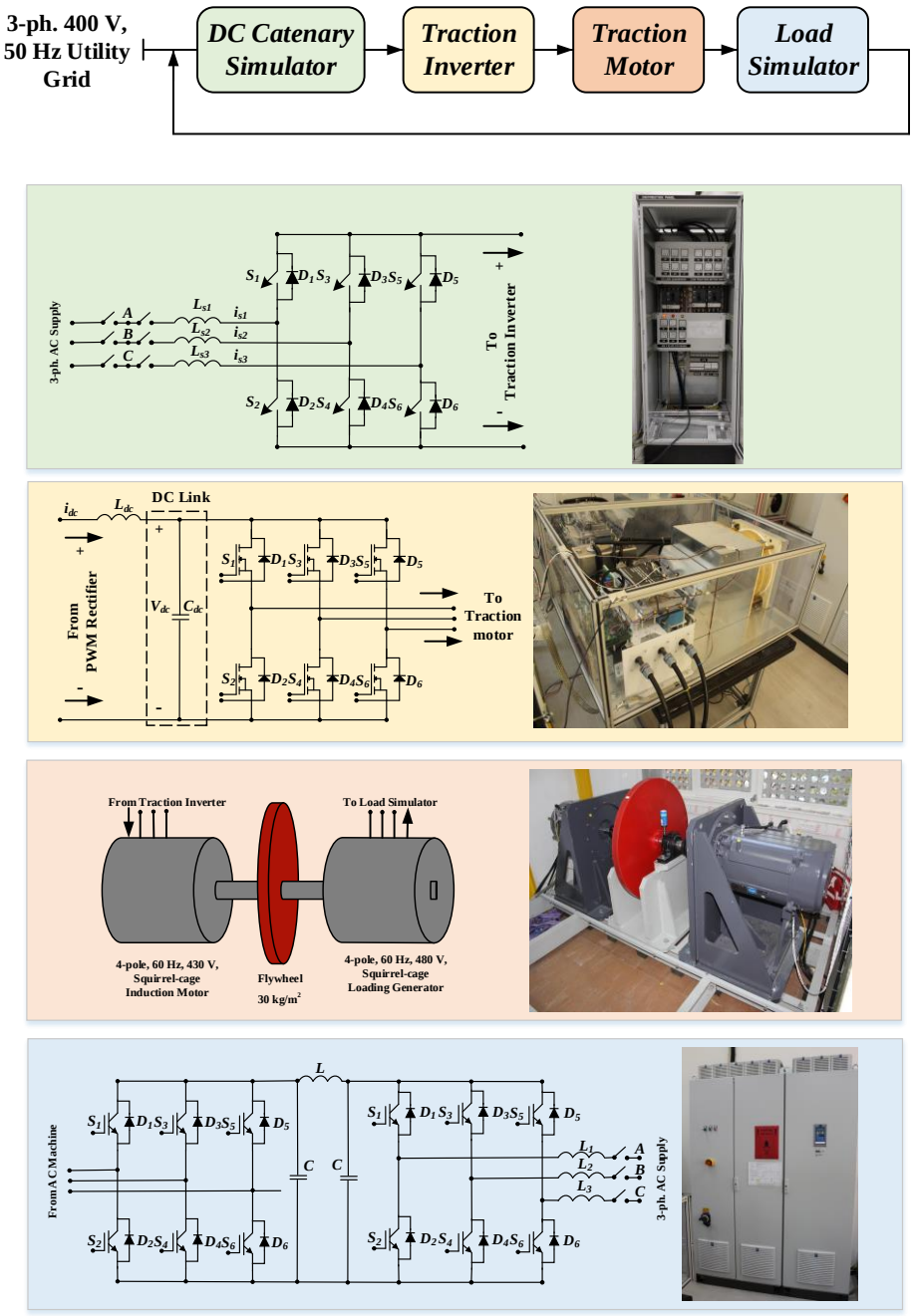


Figure 6.2: Traction System Laboratory Prototype for Railway Transportation, and the Block Diagrams of the System

On the other hand, because the PWM rectifier comes from a three-phase 400 V grid, due to the diode bridge rectification feature of the PWM rectifier, the lowest permanent voltage is 565 V DC. The PWM rectifier output voltage is kept constant at 650 V DC in this article to compare SHE-PWM and SVM-PWM techniques. The

traction inverter can deliver 180 kVA to the traction motor. Simultaneously, the catenary voltage ranges between 750 and 900 V. However, the inverter's output should be reduced linearly from 180 to 88 kVA at 565 V for catenary voltages of less than 750 V. The traction motor sets stated technical specifications in Table 6.1 are given in Figure 6.2. According to the load generator, a load simulator named VEMo drive has been chosen, and this simulator is shown in Figure 6.2.

Table 6.1:Technical Properties Motor-Generator Set

	4-Pole, 60 Hz, 475 V, 125 kW, 0.81 pf, eff =93.5%	
Traction Motor (VEM-DKCBZ 0212-4)	Rated Torque Maximum Torque Rated Speed Maximum Speed	671 Nm 940 Nm 1779 rpm 5700 rpm
	4-Pole, 60 Hz, 430 V, 130 kW, 0.82 pf, eff =93.7%	
Loading Machine (VEM-DKoBZ 0610-4)	Rated Torque Maximum Torque Rated Speed Maximum Speed	700 Nm 1200 Nm 177rpm 5500 rpm
Moment of inertias	Motor inertia Generator inertia Flywheel inertia	0.67 kg-m^2 0.84 kg-m^2 30 kg-m^2

In Figure 6.3, SHE-PWM control scheme is represented. In the experiments, speed control is based for V/f control and modulation index. As mentioned before chapters' modulation index is crucial to determine lookup table (LUT) dependent to motor operating points. TMS320F 28377 produced by Texas Instruments is programmed for sending gate signals to Cree 1700 V SiC gate driver circuits.

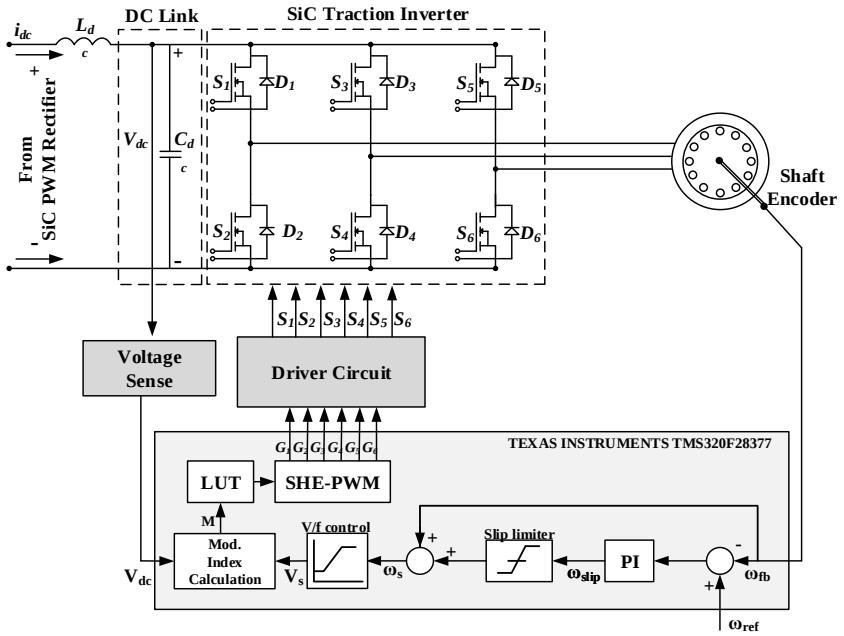


Figure 6.3: Control Scheme of SHE-PWM SiC Traction Inverter

The operating points of the experiments are presented in Figure 6.4. In this torque-speed graph, A, B, C, and D points symbolized different torque and speed values are marked. Experiments are realized at 565 V and 750 V DC-Link voltages, and different inverter output working points A, B, and C calculated with HHO algorithm. A defines full-load at 60 Hz frequency. Similarly B and C identfy full-load at 30 Hz, and half-load at 60 Hz.

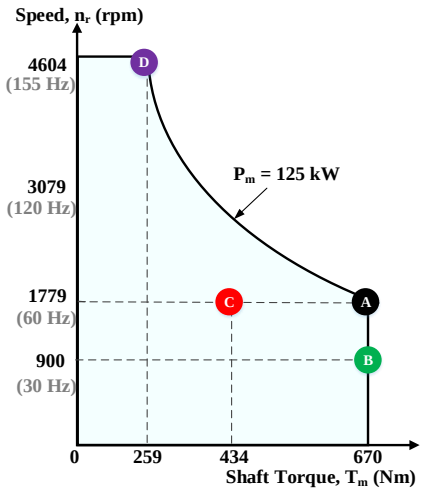


Figure 6.4: Speed Versus Torque Diagram for the Sample Traction Motor

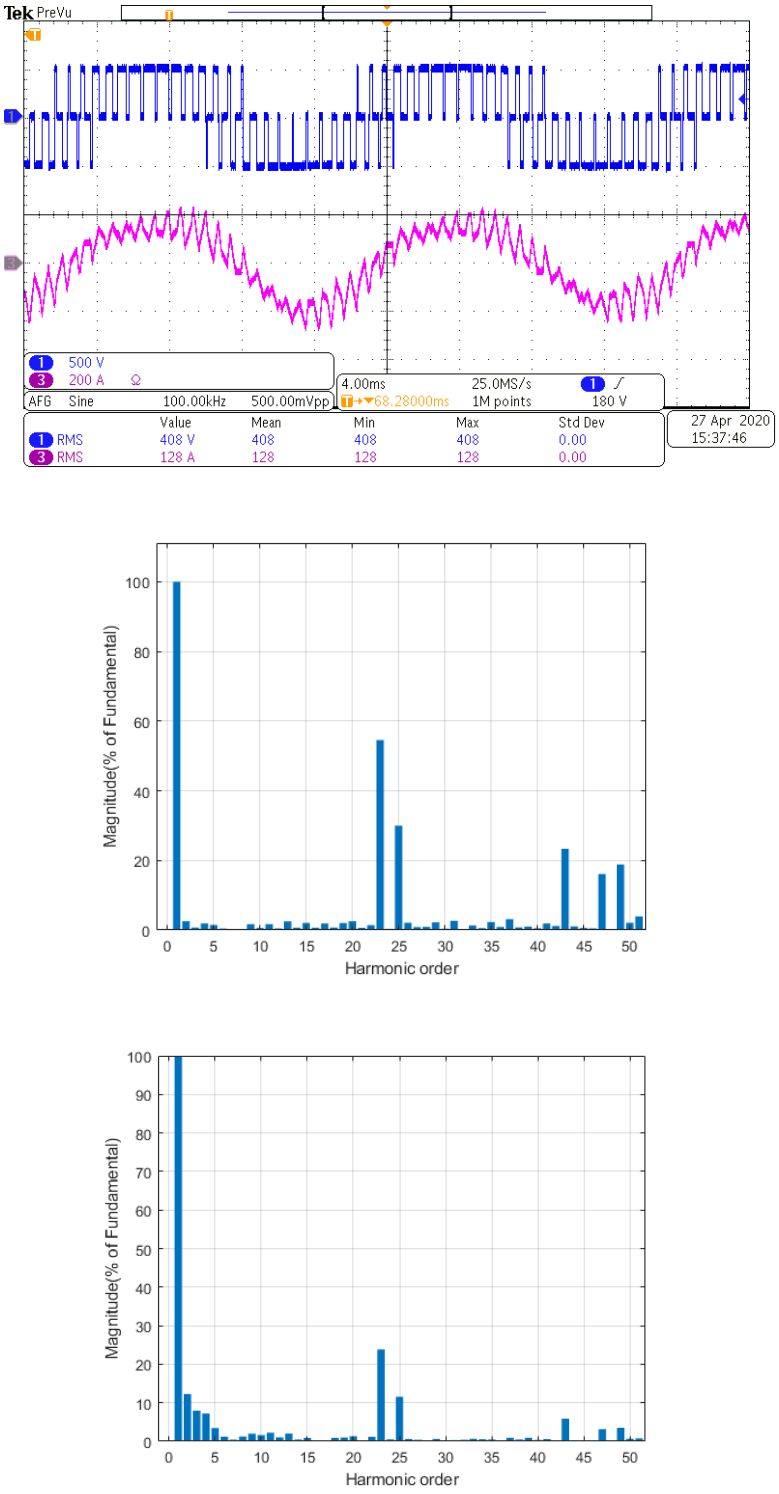


Figure 6.5: Inverter Output Voltage and Current Waveforms Obtained Switching Angles by HHO and Their FFT Distributions at 60 Hz, Full-Load at Point A, 565 V DC-Link Voltage

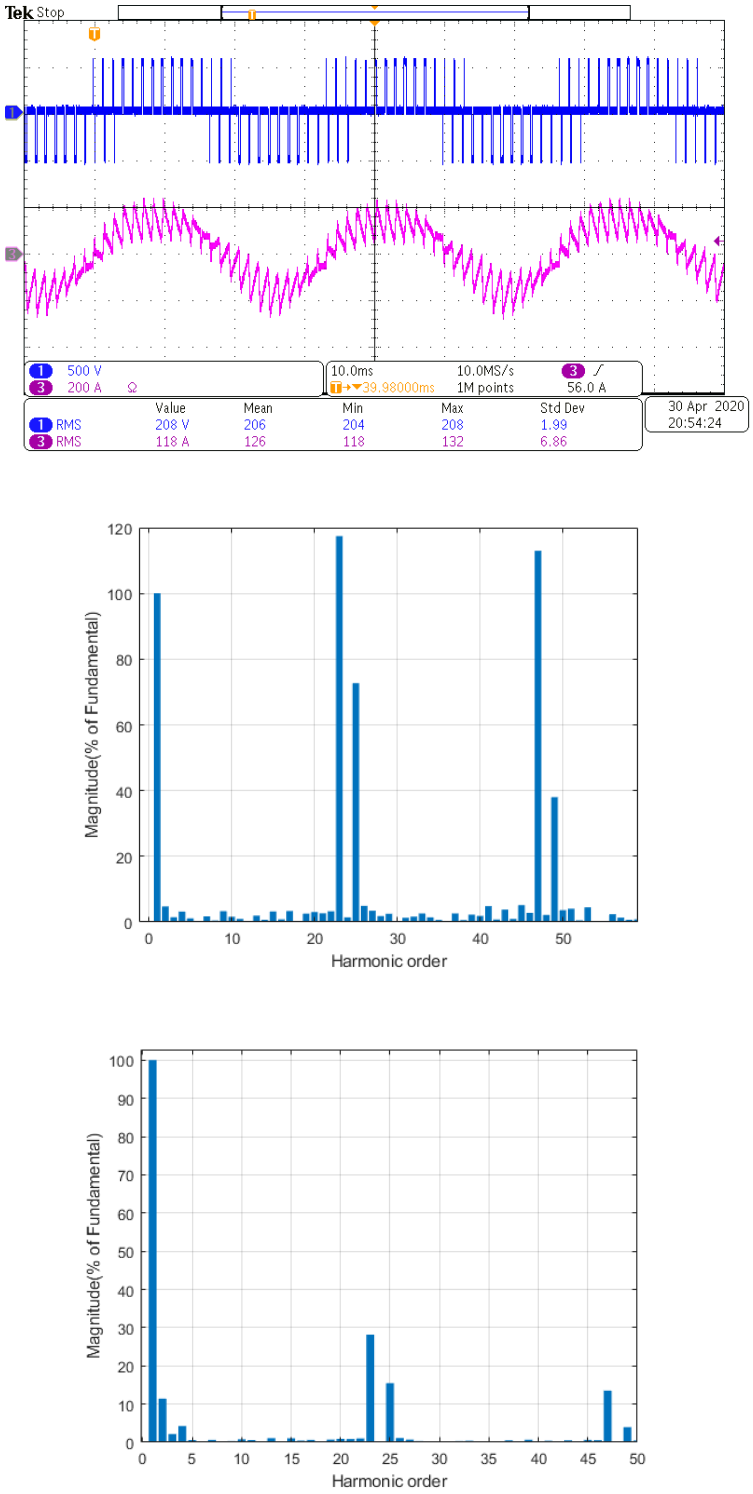


Figure 6.6: Inverter Output Voltage and Current Waveforms Obtained Switching Angles by HHO and Their FFT Distributions at 30 Hz, Full-Load at Point B, 565 V DC-Link Voltage

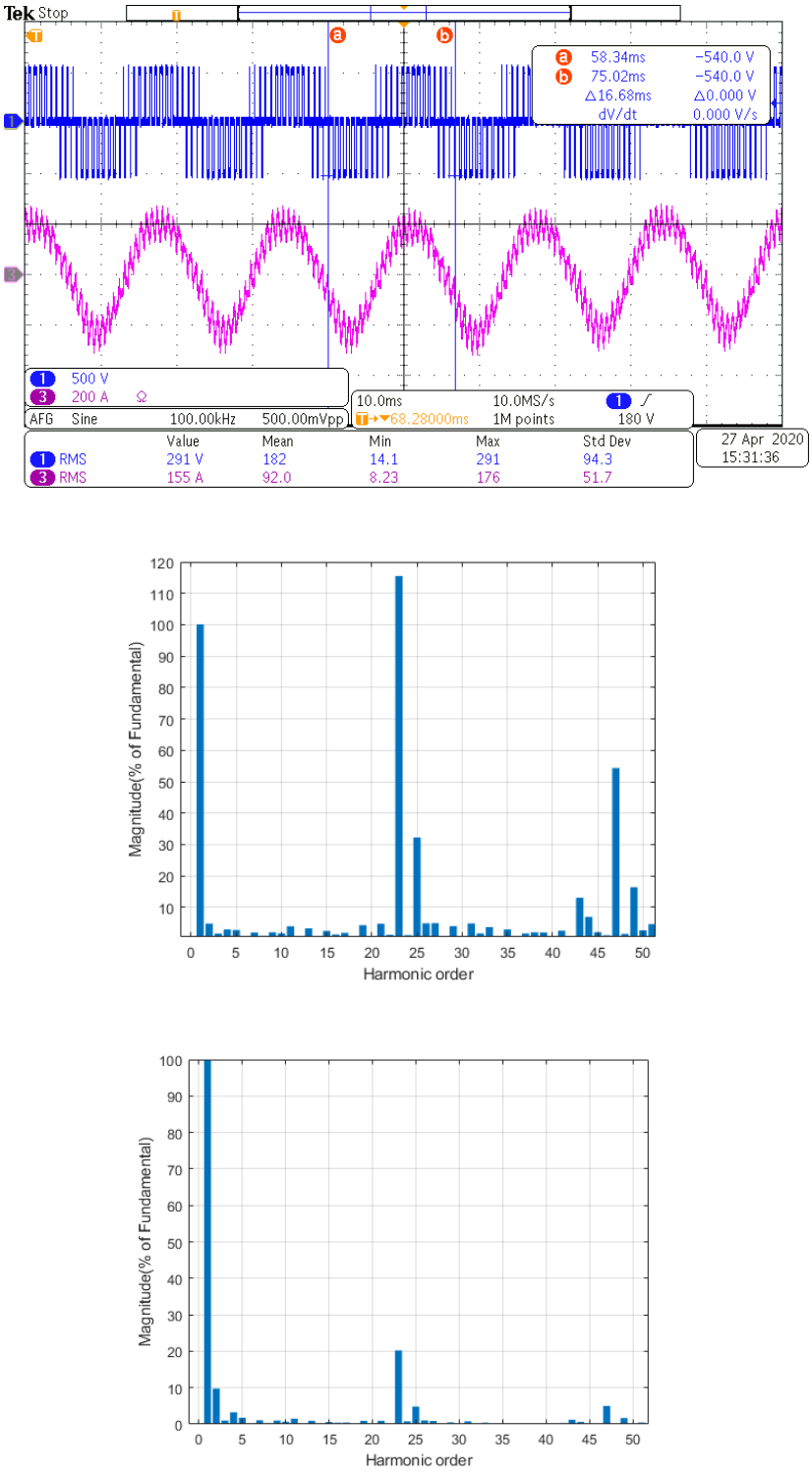


Figure 6.7: Inverter Output Voltage and Current Waveforms Obtained Switching Angles by HHO and Their FFT Distributions at 60 Hz, Half-Load at C point, 565 V DC-Link Voltage

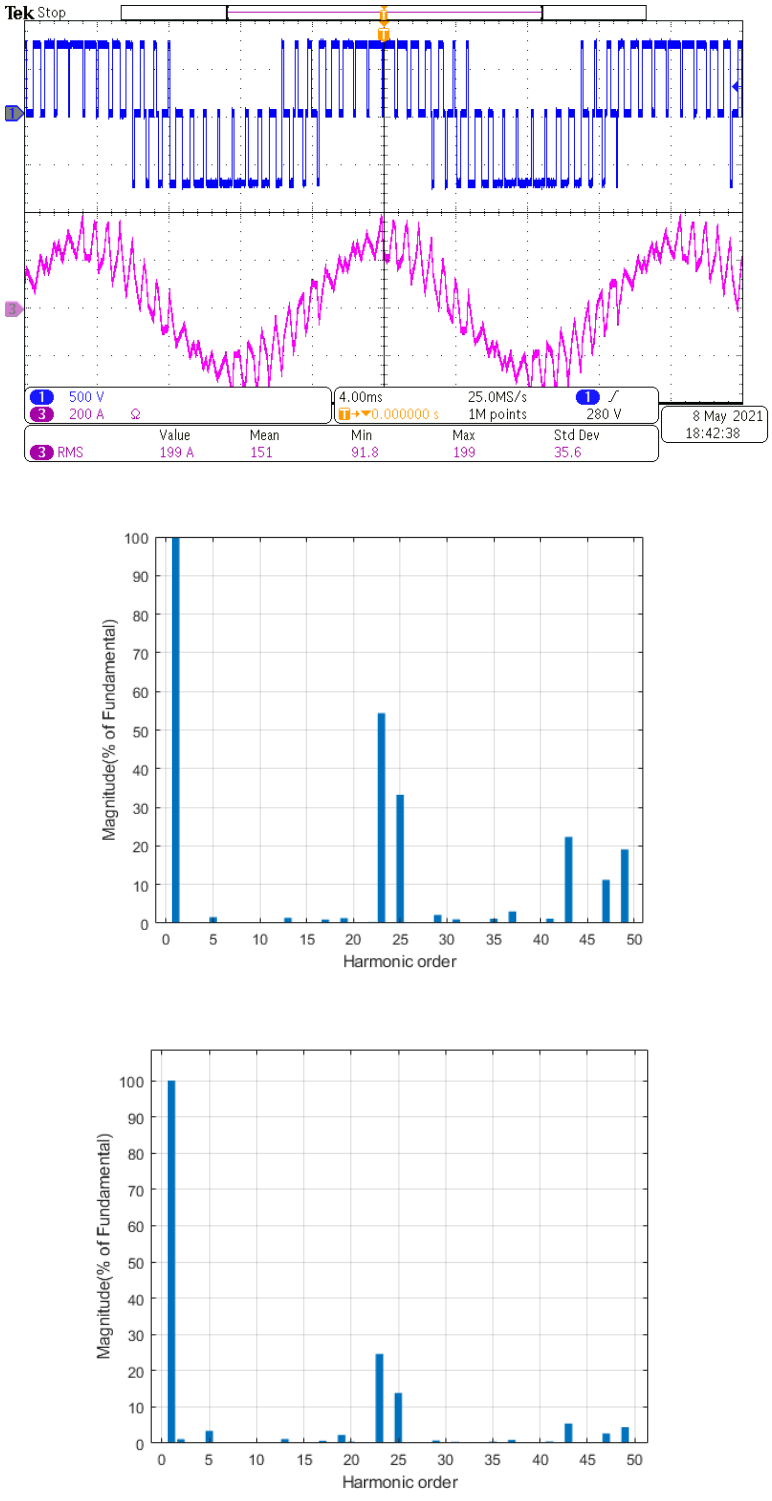


Figure 6.8: Inverter Output Voltage and Current Waveforms Obtained Switching Angles by HHO and Their FFT Distributions at 60 Hz, Full-Load at A point, 750 V DC-Link Voltage

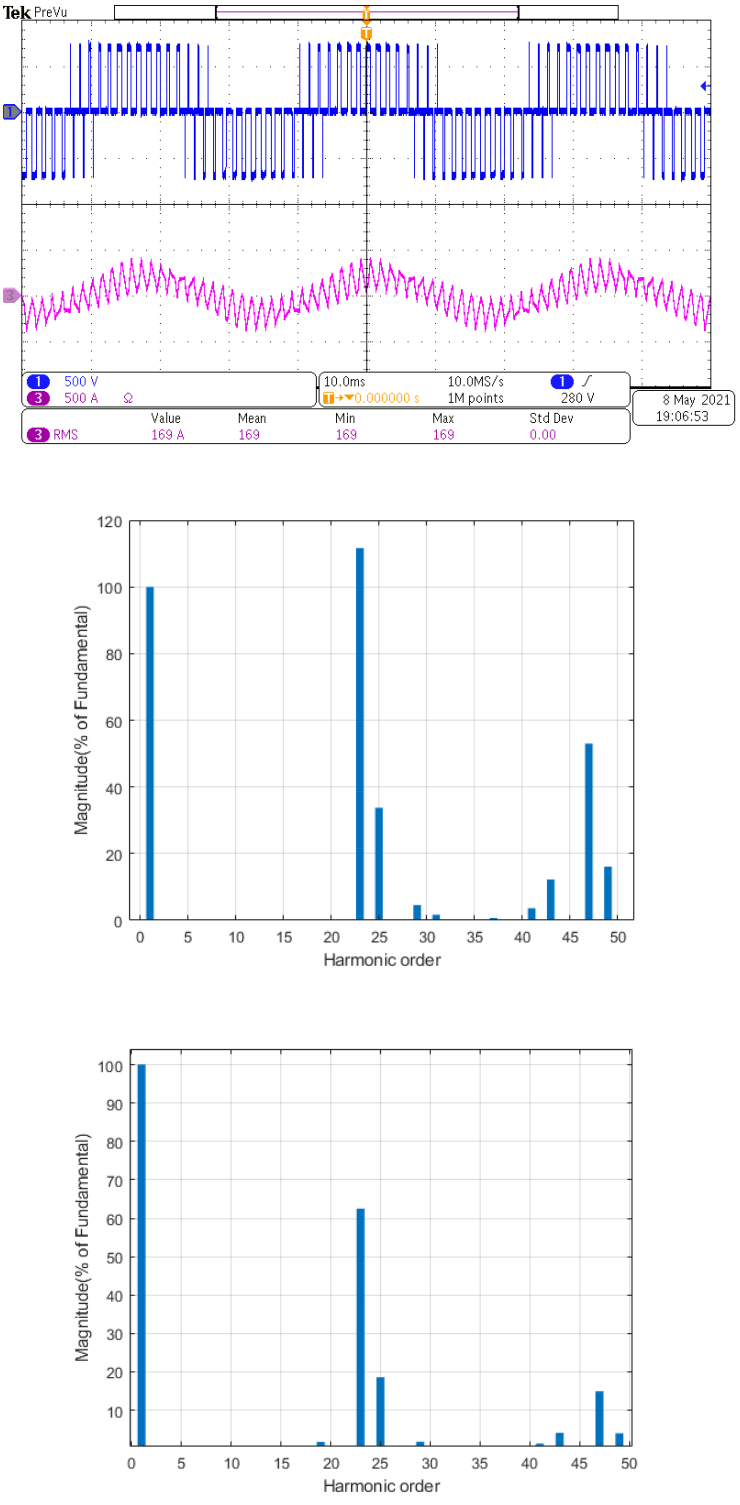


Figure 6.9: Inverter Output Voltage and Current Waveforms Obtained Switching Angles by HHO and Their FFT Distributions at 30 Hz, Full-Load at B point, 750 V DC-Link Voltage

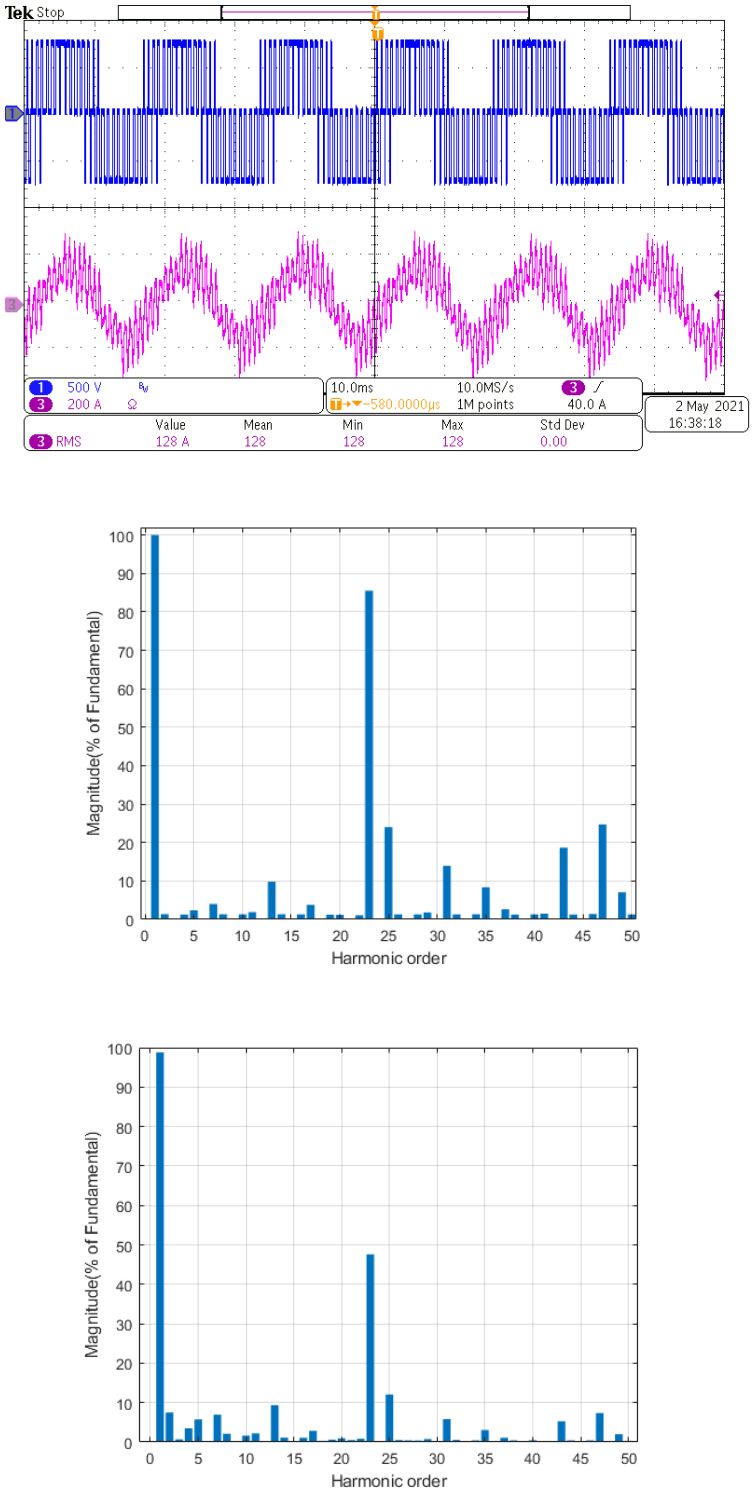


Figure 6.10: Inverter Output Voltage and Current Waveforms Obtained Switching Angles by HHO and Their FFT Distributions at 60 Hz, Half-Load at C point, 750 V DC-Link Voltage

Figure 6.5 shows the experiment at A point implemented to SHE-PWM switching angles calculated by HHO at 565 V DC-Link voltage. In TMS28377 implementation, problems such as incomplete calculated angle values, missing decimal angles, memory, and speed problems are encountered. Thus, the desired harmonics were not eliminated, but their magnitudes were low. Furthermore, Figure 6.6 and Figure 6.7 present the results of B and C points at the same voltage. It is seen that low-order harmonics magnitudes are low without eliminating.

Similarly, in Figure 6.8, the experiment at A point was carried out according to the SHE-PWM switching angles calculated by HHO at 750 V DC-Link voltage. In Figures 6.8 and 6.9, voltage and current FFT waveforms have eliminated low-order harmonics with the code improvements. In these Figures, the THD of voltage and current signals are small. But in Figure 10, there are low-order harmonics with small magnitudes because of lower torque value.

CHAPTER 6

DISCUSSION

The most common modulation techniques used in the design of device commutated DC / AC converters are Sinusoidal PWM (SPWM) and Space Vector PWM (SVPWM). For these modulation techniques, the minimum carrier signal frequency, fc(min), is recommended to be $21f_1$ in the literature [144] where, f_1 is the fundamental frequency of applied stator voltages.

Due to the relative ease with which harmonic voltages can be filtered at high frequencies, it is desirable to use as high a switching frequency as possible, except for one significant disadvantage: switching losses in the inverter switching devices increase proportionally to the switching frequency fsw. As a result, in the majority of applications, the switching frequency is either less than 6 kHz or greater than 20 kHz in order to stay above the audible range. If the optimal switching frequency is found to be between 6 and 20 kHz (based on overall system performance), the disadvantages of increasing it to 20 kHz are frequently outweighed by the benefit of no audible noise at f_{sw} of 20 kHz or greater.

Thus, in 50-60 Hz applications such as ac drive systems (where the fundamental frequency of the inverter may be as high as 20 kHz), the frequency modulation ratio m_f may be as low as 9 for switching frequencies less than 2 kHz. On the other hand, for switching frequencies greater than 20 kHz, mf will be greater than 100. How large m_f is dictates the desirable relations between the triangular waveform signal and the control signal. m_f =21 is treated as the boundary line between large and small in this discussion, though its selection is rather arbitrary. The amplitude modulation ratio m_a is assumed to be less than one in this case. However, for SHEM $f_{c(min)}=(2N+1)\,f_1$ where, N is the number of SHEM angles to be determined. N angles are used to fix the fundamental component of the line-to-neutral voltage waveform

and to eliminate *N*-1 low-order odd harmonics such as 5^{th}, 7^{th}, 11^{th}, 13^{th}, 17^{th}, 19^{th}, and etc., according to TLN1 technique. In comparison with SHEM-PWM, switching frequencies *fsw* = *fc* for SPWM and SVPWM are given in Table 7.1. Table II gives detailed information about switching frequency and harmonics to be eliminated as a function of number of SHEM angles *N* and fundamental frequency of applied stator voltages f_1 at operating points A, B, C and D in Figure 7.1.

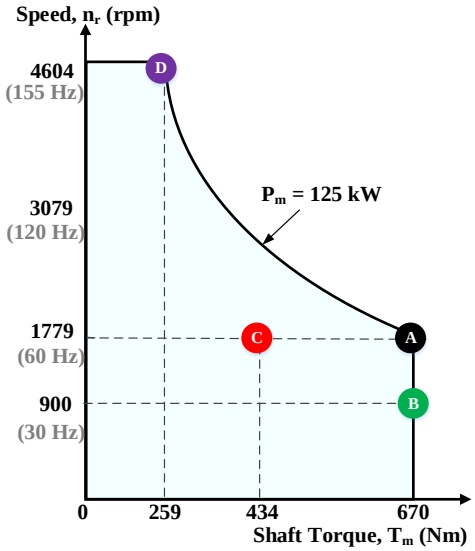


Figure 7.1: Speed Versus Torque Diagram for the Sample Traction Motor

Table 7.1: Modulation Index Values for SPWM, SVPWM, and SHEM function of V_{dc} = 750 V

		Modulation index, M			
Op. Point	**f_1 Hz**	**$\frac{2\widehat{V_1}}{V_{dc}}$ for SPWM**	**$\frac{\sqrt{3}\widehat{V_1}}{V_{dc}}$ for SVPWM**	**$\frac{\pi\widehat{V_1}}{2V_{dc}}$ for SHEM**	**V_1 l-to-n Volt**
A	60	1.05	0.91	0.82	277
B	30	0.51	0.45	0.41	138
C	60	1.05	0.91	0.82	277
D	155	1.05	0.91	0.82	277

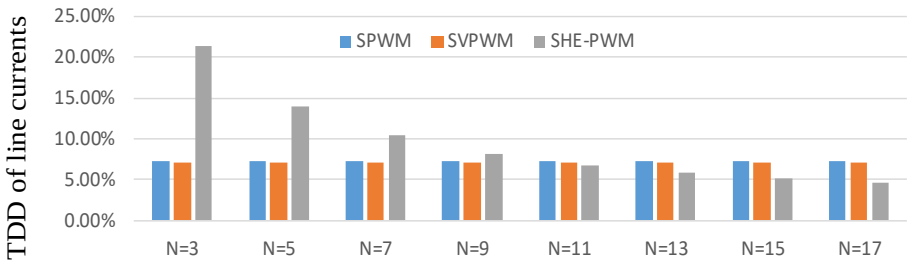
Table 7.2: Switching Frequencis Againist the Number of SHEM Angles for Three Different Fundamental Frequencies

No of SHEM angles, *N*	**Harmonics to be eliminated**	**Frequency of fundamental stator voltage, f_1**		
		30 Hz	**60 Hz**	**155 Hz**
		Switching frequency, $f_{sw} = (2N+1)f_1$ Hz		
3	5, 7	210	420	1085
5	5, 7, 11, 13	330	660	1705
7	5, 7, 11, 13, 17, 19	450	900	2325
9	5, 7, 11, 13, 17, 19, 23, 25	570	1140	2945
11	5, 7, 11, 13, 17, 19, 23, 25, 29, 31	690	1380	3565
13	5, 7, 11, 13, 17, 19, 23, 25, 29, 31, 35, 37	810	1620	4185
15	5, 7, 11, 13, 17, 19, 23, 25, 29, 31, 35, 37, 41, 43	930	1860	4805
17	5, 7, 11, 13, 17, 19, 23, 25, 29, 31, 35, 37, 41, 43, 47, 49	1050	2100	5425

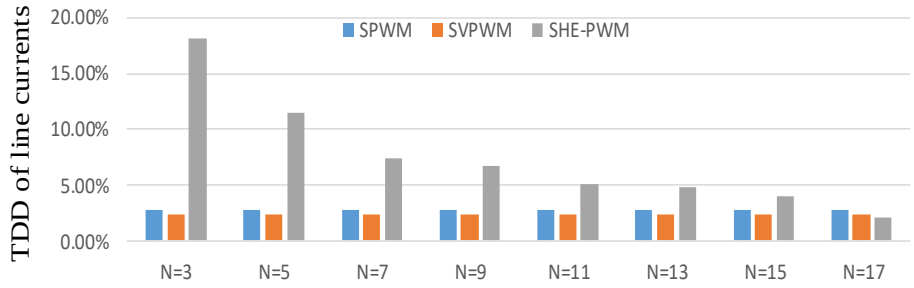
Table 7.3: Minimum Pulse Durations at Operating Points A, B, and D

		Operating points		
Modulation Techniques	**f_{sw} Hz**	**Point A 60 Hz**	**Point B 30 Hz**	**Point D 155 Hz**
SPWM	3255	2.5 µs	1 µs	8 µs
SVPWM	3255	8 µs	7 µs	7 µs
SHEM(N=7)	2100	85 µs	165 µs	35 µs
SHEM(N=17)	5450	40 µs	80 µs	14.5 µs

Three-phase line-to-neutral and line-to-line voltage waveforms synthesized by SPWM, SVPWM, and SHEM are applied to the stator terminals of the traction motor. THD of these waveforms is high, i.e., around 90% owing to multiples switching frequency harmonics and their sidebands in SPWM and SVPWM waveforms. However, in SHEM, all predefined low-order harmonics are moved to frequencies above the highest eliminated harmonic frequency. At such high harmonic frequencies, the input impedance of the traction motor is much higher than the input impedance of the motor at the fundamental frequency. Therefore, each high-frequency harmonic component can drive a relatively small current through the stator. It means that TDD of the line current waveform are expected to be much smaller than THD of applied stator voltages e.g. from a few percent to twenty percent. In order to illustrate effects of fundamental frequency component, the number of SHEM angles, switching frequency and applied PWM method on TDD of motor line currents characteristics in Figure 7.2 are obtained. Since the traction motor drive operates in a wide frequency and torque range as given in Figure 7.1, TDD performances of the three PWM techniques are compared in Figure 7.3 for operating points A to D. The minimum pulse widths according to the operating points are shown in the Table 7.3. This table is important for deciding the switching devices turn on and turn off times. SHE-PWM has the longest switching time and provides flexibility in switch selection.

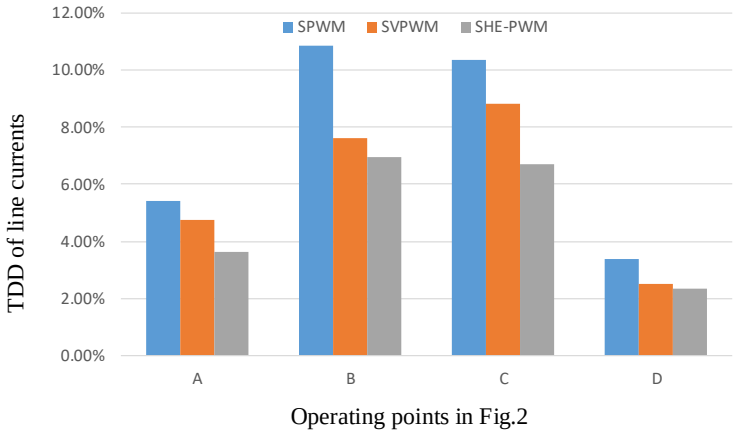


(a)

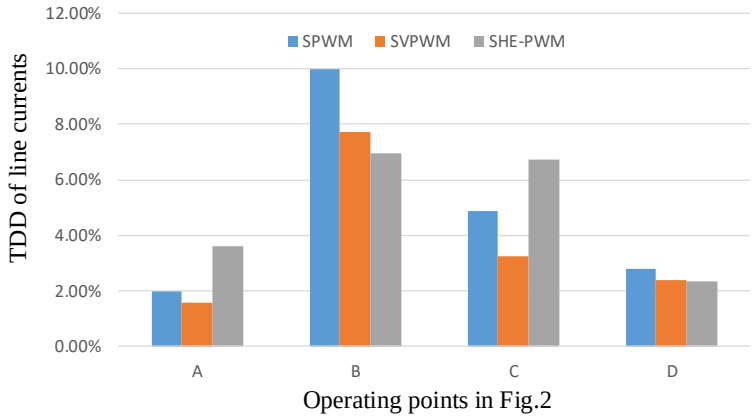


(b)

Figure 7.2: Comparison of SPWM, SVPWM and SHEM-PWM in view of TDD variations of motor line currents, (a). At operating point A, $f_{sw} = 21f_1 = 1260$ Hz for SPWM, SVPWM and $f_{sw} = (2N+1)60$ Hz for SHEM; (b). At operating point D, $f_{sw} = f_c = 3255$ Hz for SPWM, SVPWM and $f_{sw} = (2N+1)155$ Hz for SHEM



(a)



(b)

Figure 7.3: Comparison of SPWM, SVPWM and SHEM-PWM in view of TDD variations of motor line currents at operating points A, B, C and D, (a). $f_{sw} = 21f_1$ Hz for SPWM and SVPWM, N=17 for SHEM, (b). constant $f_{sw} = f_c = 3255$ Hz for SPWM and SVPWM, N=17 for SHEM

Another important point in the design of the traction converter is the minimum pulse width. Minimum pulse width should be greater than the sum of turn-on time, turn-off time, delay times and dead band. Following conclusions can be drawn from Figure 7.2 and 7.3 and Tables 7.1 and 7.2.

i. Higher switching frequency lowers TDD of currents.

ii. SHEM with $N \geq 9$ can compete with SPWM and SVPWM, if $f_{sw} = 21\ f_1$.

iii. SPWM requires over modulation even for the rated DC-Link voltage and therefore is not suitable for traction motor speeds at and over the rated value.

iv. SVPWM fits well with this variable speed application. $f_{sw} = 3255$ Hz may require the selection of an overrated IGBT.

v. SHEM with N=17 fits well with this medium power, variable speed application but requires the use of SiC power MOSFETs as the major switching device.

vi. Highest values of minimum pulse width are obtained for SHEM in comparison with other PWM techniques. This property provides us

flexibility in the implementation. In the case of IGBT-based SVPWM applications the designers are forced to delete the narrow pulses, thus causing an increase in TDD.

vii. By increasing the motor's inductance, the harmonics' amplitudes are reduced, and the total THD and TDD values are decreased.

viii. By increasing the motor frequency, the SHE-PWM switching frequency is increased, and the lower order harmonics are dragged up to higher harmonic levels. As a result, the motor current will be closer to a pure sinusoidal signal.

CHAPTER 7

CONCLUSION

Variable Frequency Drives (VFDs) have become a more reliable and cost-effective speed control technique due to technological advancements. Due to their improvements, VFDs have grown into potent digital microprocessor processors and high-frequency power devices in recent years. SHEM is a switching method that simultaneously controls the fundamental output voltage of the inverter and eliminates specified lower-order harmonics 5^{th}, 7^{th}, 11^{th}, 13^{th}, $17^{th,}$ and 19^{th} by intentionally introducing notches in the waveform in 565 and 750 V DC-Link voltages and 30 and 60 Hz operating frequencies. Due to the low switching frequency required by this method, switching losses are minimized, which is critical for the efficiency of high-power converters.

Until recently, the only way to utilize SHEM for inverters was to find these solutions in advance and store them as lookup tables in the microcontrollers. However, offline application of SHEM requires discretized modulation index values, a large lookup table, and a careful examination of the solution space in which no feasible solution to the SHEM equations can be found. These disadvantages of SHEM's offline application have prompted researchers to seek solutions. With the progression of ever-increasing computing power, emerging microcontroller technologies' power and speed and their implementation. It has become possible to run complex search and optimization algorithms in real-time. It is not necessary to use these fast and powerful microcontrollers. It is prohibitively difficult to implement an efficient, robust, dynamic, and reliable power conversion system.

Compared to SHEM with SPWM and SVPWM methods, SHEM has advantages at low frequencies such as eliminated desired harmonics, lower THD, low (up to 2100 Hz, 50^{th} harmonics for 60 Hz motor frequency). On the other hand, there are also

significant disadvantages. Due to the nonlinear nature of SHEM equations, determining the switching angle set that eliminates the specified harmonics becomes more difficult as the number of harmonics increases. Numerous iterative and metaheuristic algorithms for solving SHEM equations have been investigated in the literature. In this thesis, PSO, GWO, WOA, and HHO metaheuristic algorithms are examined and compared. Each method successfully locates the correct solution within their respective reference frames; because the modulation index is also variable, many solution sets should be found for applications with variable DC link voltage. But HHO solves the SHEM problem with a high convergence rate and accuracy.

SHEM equations have been solved in real-time using GPUs in this thesis work for switching angles that eliminate desired harmonics increase in the output voltage of a traction inverter when the DC link input is variable voltage. The PSO, GWO, WOA, and HHO algorithms were implemented as a module and online in the GPUs. SHEM's application was successfully validated using CUDA programming of the computer and the hardware between the computer and the GPU module. New switching angles can be created using the parameters in the HHO module to be discovered in a total of 80 microseconds. As a result, at this point, only six harmonics could be eliminated online. Selection by increasing the number of optimized parameters, rewriting inefficient portions of the GPU code, and utilizing a more precise random number generator may result in a shorter solution.

Although the simulation results for the entire system can be considered successful, several significant points were made during the experimental work for future research. It was observed that one of the phase currents had accumulated to extremely high values, either positively or negatively. This issue could result from faulty switching under normal circumstances; however, the switching waveforms were remarkably similar, examined for potential problems, and implemented necessary countermeasures in the TMS28377 code.

As can be seen from the experimental results, a motor inductor is critical to achieving optimal performance for VSIs. The inverter's output inductance, motor inductance two µH, was observed to sink low-order harmonics caused by nonlinear loads coupled to a common point of coupling, such as standby power supplies. The switching frequency must be increased by increasing the switching angles in the output voltage of the inverter the inverter's waveform, some of which may be eliminated online. It is critical to design the inductance value according to published guidelines to inject a nearly sinusoidal output current that complies with standards while maintaining a low.

In the future, GPU units will be applied to the motor drive system by adding ADC, DAC, and extra modules. The SHEM will be used online to traction inverters with a variable DC link voltage obtained in real-time by implementing the HHO algorithm in the GPU module. This group's mentality will be other topologies and applications, such as multilevel inverters and STATCOMs, are amenable to SHEM application.

REFERENCES

[1] Cai, W., Wu, X., Zhou, M. *et al.* 2021. Review and Development of Electric Motor Systems and Electric Powertrains for New Energy Vehicles. Automot. Innov. 4, pp. 3–22.

[2] Saidur, R., 2010. A review on electrical motors energy use and energy savings. Renewable and Sustainable Energy Reviews, 14(3), pp. 877-898, doi.org/10.1016/j.rser.2009.10.018.

[3] Koch, A., Bürchner, T., Herrmann, T., Lienkamp, M., 2021. Eco-Driving for Different Electric Powertrain Topologies Considering Motor Efficiency, 12(1), doi.org/10.3390/wevj12010006.

[4] https://www.iea.org/reports/energy-efficiency-policy-opportunities-for-electric-motor-driven-systems

[5] Bose, B. K., 1993. Variable frequency drives-technology and applications, ISIE'93 - Budapest: IEEE International Symposium on Industrial Electronics *Conference Proceedings*, pp. 1-18, doi: 10.1109/ISIE.1993.268822.

[6] Baliga, B. J., 1994. Power semiconductor devices for variable-frequency drives, in Proceedings of the IEEE, vol. 82, no. 8, pp. 1112-1122, doi: 10.1109/5.301680.

[7] Slemon, G. R., 1994. Electrical machines for variable-frequency drives, in Proceedings of the IEEE, vol. 82, no. 8, pp. 1123-1139, doi: 10.1109/5.301681.

[8] Khalid, N., 2014, Efficient Energy Management: Is Variable Frequency Drives the Solution, Procedia - Social and Behavioral Sciences, 145, 371-376, doi.org/10.1016/j.sbspro.2014.06.046.

[9] Agamloh, E. B., 2017. Power and Efficiency Measurement of Motor-Variable-Frequency Drive Systems, IEEE Transactions on Industry Applications, vol. 53, no. 1, pp. 766-773, doi: 10.1109/TIA.2016.2602807.

[10] Pan, J., Ke, Z., Na, R., Zhang, Z., and Xu, L., 2019. High-speed and High-dynamic Variable Frequency Drive Using Modular Multilevel Converter and SiC Devices, 2019 IEEE Applied Power Electronics Conference and Exposition (APEC), pp. 2627-2631, doi: 10.1109/APEC.2019.8721903.

[11] https://www.semikron.com/innovation-technology/igbt-generation-7.html

[12] https://www.infineon.com/cms/en/product/power/mosfet/silicon-carbide/

[13] Prashanth B. U. V., Ahmed M. R., .2021. FPGA Implementation of Bio-inspired Computing Based Deep Learning Model. In: Tripathy A., Sarkar M., Sahoo J., Li KC., Chinara S. (eds) Advances in Distributed Computing and Machine Learning. Lecture Notes in Networks and Systems, vol 127. Springer,

[14] Ayachi, R., Said, Y. & Ben Abdelali, A. 2021. Optimizing Neural Networks for Efficient FPGA Implementation: A Survey. Arch Computat Methods Eng.

[15] Oberkirsch, L., Quinto, D. M., Schwarzbözl, P., Hoffschmidt, B. 2021. GPU-based aim point optimization for solar tower power plants, Solar Energy, 220, 1089-1098.

[16] Oh, K., Jung, K., 2004. GPU implementation of neural networks, Pattern Recognition, 37(6), 1311-1314.

[17] Zhou, Y., Tan, Y., 2009. GPU-based parallel particle swarm optimization, 2009 IEEE Congress on Evolutionary Computation, 2009, pp. 1493-1500.

[18] Preis, T., Virnau, P., Paul, W., Schneider, J. J., 2009. GPU accelerated Monte Carlo simulation of the 2D and 3D Ising model, Journal of Computational Physics, 228(12), 4468-4477.

[19] Xin, H., Yapeng, Y., Zhiwen, C., Jianhua, S., Hao, C., 2021. Efficient parallel A* search on multi-GPU system, Future Generation Computer Systems, 123, 35-47.

[20] Tang, J., Yang, Y., Blaabjerg, F., Chen, J., Diao, L., Liu, Z., 2018. Parameter Identification of Inverter-Fed Induction Motors: A Review, Energies, 11(9).

[21] Von Jouanne, A., Enjeti, P., Gray, W. 1996. Application issues for PWM adjustable speed AC motor drives, IEEE Industry Applications Magazine, vol. 2, no. 5, pp. doi: 10.1109/2943.532149.

[22] Zhang, Z., Hu, Y., Chen, X., Jewell, G. W., Li, H., 2021. A Review on Conductive Common-Mode EMI Suppression Methods in Inverter Fed Motor Drives, in IEEE Access, vol. 9, pp. 18345-18360, doi: 10.1109/ACCESS.2021.3054514.

[23] Rajan, R., Fernandez, F. M., Yang, Y., 2021. Primary frequency control techniques for large-scale PV-integrated power systems: A review, Renewable and Sustainable Energy Reviews, 144.

[24] Athari, H., Niroomand, M., Ataei, M. 2017. Review and Classification of Control Systems in Grid-tied Inverters, Renewable and Sustainable Energy Reviews, 72, 1167-1176.

[25] Niroomand, M., Karshenas, H. R., 2010. Review and comparison of control methods for uninterruptible power supplies, 2010 1st Power Electronic & Drive Systems & Technologies Conference (PEDSTC), pp. 18-23, doi: 10.1109/PEDSTC.2010.5471864.

[26] Bekiarov, S. B., Emadi, A., 2002. Uninterruptible power supplies: classification, operation, dynamics, and control, APEC. Seventeenth Annual IEEE Applied Power Electronics Conference and Exposition (Cat. No.02CH37335), pp. 597-604 vol.1, doi: 10.1109/APEC.2002.989305.

[27] Aamir, M., Kalwar, K. A., Mekhilef, S., 2016. Review: Uninterruptible Power Supply (UPS) system, Renewable and Sustainable Energy Reviews, 58, 1395-1410.

[28] Hassanpoor, A., Norrga, S., Nee, H., and Ängquist, L., 2012. Evaluation of different carrier-based PWM methods for modular multilevel converters for HVDC application, IECON 2012 - 38th Annual Conference on IEEE Industrial Electronics Society, pp. 388-393, doi: 10.1109/IECON.2012.6388789.

[29] Alkhawaldeh, L., Bhuiya, M. A., Eid, M., Youssef, M. Z., 2021. An Enhanced EPP-MPPT Algorithm With Modified Control Technique in Solar-Based Inverter Applications: Analysis and Experimentation, IEEE Access, vol. 9, pp. 8158-8166, doi: 10.1109/ACCESS.2021.3049517.

[30] Sarker, R., Datta, A., Debnath, S., An Improved Multicarrier PWM (MCPWM) Technique with a New Harmonic Mitigation Strategy for Cascaded H-Bridge Multilevel Inverter Applications, in IEEE Transactions on Industrial Informatics, doi: 10.1109/TII.2021.3087458.

[31] Fong, Y. C., Cheng, K. W. E., Raghu Raman, S., A Modular Concept Development for Resonant Soft-Charging Step-Up Switched-Capacitor Multilevel Inverter for High-frequency AC Distribution and Applications, in IEEE Journal of Emerging and Selected Topics in Power Electronics, doi: 10.1109/JESTPE.2020.3043126.

[32] Jayakumar, V., Sachinamreiss, G. N., Karthikeyan, B., 2021. Investigation of asymmetric concept on voltage source multilevel inverter for industrial applications, Materials Today: Proceedings, 45(2), 1603-1608.

[33] https://circuitdigest.com/tutorial/different-types-of-inverters

[34] Hamman, J., van der Merwe, F. S., 1988. Voltage harmonics generated by voltage-fed inverters using PWM natural sampling, in IEEE Transactions on Power Electronics, vol. 3, no. 3, pp. 297-302, doi: 10.1109/63.17947.

[35] McGrath B. P., Holmes, D. G., 2009. A General Analytical Method for Calculating Inverter DC-Link Current Harmonics, in IEEE Transactions on Industry Applications, vol. 45, no. 5, pp. 1851-1859, doi: 10.1109/TIA.2009.2027556.

[36] Santos, R., Gonçalves, F.A.S., 2021. Sinusoidal PWM Techniques Comparison for the Quasi-Y-Source Inverter. J Control Autom Electr Syst 32, 1399–1407.

[37] Y. Chen et al., 2016. A ZVS Grid-Connected Full-Bridge Inverter With a Novel ZVS SPWM Scheme, in IEEE Transactions on Power Electronics, vol. 31, no. 5, pp. 3626-3638, doi: 10.1109/TPEL.2015.2456032.

[38] Yao, H., Yan, Y., Shi, T., Zhang, G., Wang, Z., Xia, C., 2021. A Novel SVPWM Scheme for Field-Oriented Vector-Controlled PMSM Drive System Fed by Cascaded H-Bridge Inverter, in IEEE Transactions on Power Electronics, vol. 36, no. 8, pp. 8988-9000, doi: 10.1109/TPEL.2021.3054642.

[39] Li, Z., Guo, Y., Xia, J., Duan, Y., Zhang, X., 2020. Modified Synchronized SVPWM Strategies to Reduce Common-Mode Voltage for Three-Phase Voltage Source Inverters at Low Switching Frequency, IEEE Transactions on Industry Applications, vol. 9994, pp. 1-1, 2020.

[40] Karshenas, H. R., Kojori, H.A ., Dewan, S. B. 1995. Generalized techniques of selective harmonic elimination and current control in current source inverters/converters, in IEEE Transactions on Power Electronics, vol. 10, no. 5, pp. 566-573, doi: 10.1109/63.406844.

[41] Zhao, Z., Zhong, Y., Gao, H., Yuan, L., Lu, T., 2012. Hybrid Selective Harmonic Elimination PWM for Common-Mode Voltage Reduction in Three-Level Neutral-Point-Clamped Inverters for Variable Speed Induction Drives, in IEEE Transactions on Power Electronics, vol. 27, no. 3, pp. 1152-1158, doi: 10.1109/TPEL.2011.2162591.

[42] Konstantinou, G., Agelidis, V. G., 2014. On re-examining symmetry of two-level selective harmonic elimination PWM: Novel formulations, solutions and performance evaluation, Electric Power Systems Research, 108, 185-197.

[43] Dahidah, M. S. A., Konstantinou, G., Agelidis, V. G., 2015. A Review of Multilevel Selective Harmonic Elimination PWM: Formulations, Solving Algorithms, Implementation and Applications, in IEEE Transactions on Power Electronics, vol. 30, no. 8, pp. 4091-4106, doi: 10.1109/TPEL.2014.2355226.

[44] Kagerbauer, J. D., Jahns, T. M., 2007. Development of an Active dv/dt Control Algorithm for Reducing Inverter Conducted EMI with Minimal Impact on Switching Losses, 2007 IEEE Power Electronics Specialists Conference, pp. 894-900, doi: 10.1109/PESC.2007.4342107.

[45] Palma, L., Enjeti, P., 2002. An inverter output filter to mitigate dV/dt effects in PWM drive system, APEC. Seventeenth Annual IEEE Applied Power Electronics Conference and Exposition (Cat. No.02CH37335), pp. 550-556 vol.1, doi: 10.1109/APEC.2002.989298.

[46] Renge, M. M., Suryawanshi, H. M., 2008. Five-Level Diode Clamped Inverter to Eliminate Common Mode Voltage and Reduce dv/dt inMedium Voltage Rating Induction Motor Drives, in IEEE Transactions on Power Electronics, vol. 23, no. 4, pp. 1598-1607, doi: 10.1109/TPEL.2008.925423.

[47] Ahmadi, D., Zou, K., Li, C., Huang, Y., Wang, J., 2011. A Universal Selective Harmonic Elimination Method for High-Power Inverters, in IEEE Transactions on Power Electronics, vol. 26, no. 10, pp. 2743-2752, doi: 10.1109/TPEL.2011.2116042.

[48] Sun, J., Grotstollen, H., 1992. Solving nonlinear equations for selective harmonic eliminated PWM using predicted initial values, In: Proceedings of the international conference ind. electron., contr., instrum., automat., pp. 259–64.

[49] Sun, J., Beineke, S., Grotstollen, H., 1996. Optimal PWM based on real-time solution of harmonic elimination equations, IEEE Transactions on Power Electronics, vol. 11, no. 4, pp. 612–21.

[50] Enjeti, P. N., Ziogas, P. D., Lindsay, J. F., 1988, Programmed PWM techniques to eliminate harmonics. A critical evaluations, IEEE Industrial Applications Society, vol. 26, no. 2.

[51] Dahidah, M. S. A., Agelidis, V. G., 2008, Selective Harmonic Elimination PWM Control for Cascaded Multilevel Voltage Source Converters: A Generalized Formula, in IEEE Transactions on Power Electronics, vol. 23, no. 4, pp. 1620-1630, doi: 10.1109/TPEL.2008.925179.

[52] Wu, M., Xue, C., Li Y. W., Yang, K., 2021. A Generalized Selective Harmonic Elimination PWM Formulation With Common-Mode Voltage Reduction Ability for Multilevel Converters, in IEEE Transactions on Power

Electronics, vol. 36, no. 9, pp. 10753-10765, doi: 10.1109/TPEL.2021.3063299.

[53] Yıldırım D. et al., 2020. Full-Scale Physical Simulator of All SiC Traction Motor Drive With Onboard Supercapacitor ESS for Light-Rail Public Transportation, in IEEE Transactions on Industrial Electronics, vol. 67, no. 8, pp. 6290-6301, doi: 10.1109/TIE.2019.2934086.

[54] Nalcaci, G., Yildirim D., Ermis, M., 2020. Selective Harmonic Elimination for Light-Rail Transportation Motor Drives using Harris Hawks Algorithm, 2020 IEEE International Conference on Environment and Electrical Engineering and 2020 IEEE Industrial and Commercial Power Systems Europe (EEEIC / I&CPS Europe), pp. 1-6, doi: 10.1109/EEEIC/ICPSEurope49358.2020.9160694.

[55] Chiasson, J. N., Tolbert, L. M., Mckenzie, K. J., Du, Z., 2004. A complete solution to the harmonic elimination problem, IEEE Transactions on Power Electronics, vol. 19, no. 2, pp. 491–9.

[56] Etesami, M. H., Farokhnia, N., Fathi, S. H., 2015. Colonial competitive algorithm development toward harmonic minimization in multilevel inverters, IEEE Transactions on Industrial Informations, vol. 11, no. 2, pp. 459–66.

[57] Shen, K., Zhao, D., Mei, J., Tolbert, L., Jianze, W., Ban, M., 2014. Elimination of harmonics in a modular multilevel converter using particle swarm optimization based staircase modulation strategy, IEEE Transactions on Industrial Electronics, vol. 46, pp. 1–1.

[58] Barkati, S., Baghli, L., Madjid, E., Boucherit, M., 2008. Harmonic elimination in diode clamped multilevel inverter using evolutionary algorithms, Electric Power Systems, vol. 78, pp. 1736–46.

[59] Kumar, J., Das, B., Agarwal, P., 2009. Harmonic reduction technique for a cascade multilevel inverter, International Journal Recent Trends Engineering, vol. 1, no. 3, pp. 2–6.

[60] Hosseini Aghdam, M. G., Fathi, S. H., Gharehpetian, G. B., 2007. A complete solution of harmonics elimination problem in a multi-level inverter with

unequal DC sources, Journal of Electronics Systems, vol. 3, no. 4, pp. 259–71.

[61] Memon, M. A., Mekhilef, S., Mubin, M., Aamir, M., 2018. Selective harmonic elimination in inverters using bio-inspired intelligent algorithms for renewable energy conversion applications: A review, Renewable and Sustainable Energy Reviews, vol. 82, pp. 2235-2253.

[62] Shen, K., Zhao, D., Mei, J., Tolbert, L., Jianze, W., Ban, M., et al., 2014. Elimination of harmonics in a modular multilevel converter using particle swarm optimization based staircase modulation strategy, IEEE Trans Ind Electron, 46, pp. 1-1.

[63] Sun, J, Grotstollen H. 1992. Solving nonlinear equations for selective harmonic eliminated PWM using predicted initial values. In: Proceedings of the international conference ind. electron., contr., instrum., automat.; p. 259–64.

[64] Sun, J., Beineke, S., Grotstollen, H., 1996. Optimal PWM based on real-time solution of harmonic elimination equations, IEEE Trans Power Electron, 11 (4), pp. 612-621.

[65] Enjeti, P. N., Ziogas, P. D., Lindsay, J. F., 1988. Programmed PWM techniques to eliminate harmonics. A critical evaluations, IEEE Ind Appl Soc, 26 (2).

[66] Barkati, S., Baghli, L., Madjid, E., Boucherit, M., 2008. Harmonic elimination in diode-clamped multilevel inverter using evolutionary algorithms, Electr Power Syst Res, 78, pp. 1736-1746.

[67] Tutkun, N., 2010. Improved power quality in a single-phase PWM inverter voltage with bipolar notches through the hybrid genetic algorithms, Expert Syst Appl, 37 (8), pp. 5614-5620.

[68] Salam, Z., Soon Yee, S., Saleem, Y., 2013. On the improved computational speed and convergence of the Newton Raphson iteration method for selective harmonics elimination PWM applied to cascaded multilevel inverter with

equal and non-equal DC sources, COMPEL – Int J Comput Math Electr Electron Eng, 32 (3), pp. 901-922

[69] Sihem, G., Dib, D., Meghni, B., Zouli, M. 2017. Selective harmonic elimination strategy in eleven level inverter for PV system with unbalanced DC sources. In: AIP conference proceedings, 1814, 020008; doi: 10.1063/1.4976227.

[70] Liang, T., J., O'Connell, R. M., Hoft, R. G. 1997. Inverter harmonic reduction using Walsh function harmonic elimination method, IEEE Trans Power Electron, 12 (6), pp. 971-982.

[71] Kumar, J., Das, B., Agarwal, P. 2009. Harmonic reduction technique for a cascade multilevel inverter, Int J Recent Trends Eng, 1 (3), pp. 2-6.

[72] Kato, T. 1999. Sequential homotopy-based computation of multiple solutions for selected harmonic elimination in PWM inverters, IEEE Trans Circuits Syst I Fundam Theory Appl, 46 (5), pp. 586-593.

[73] Hosseini Aghdam, G. 2013. Optimised active harmonic elimination technique for three-level T-type inverters, IET Power Electron, 6 (3), pp. 425-433

[74] Hosseini Aghdam, M. G., Fathi, S. H., Gharehpetian, G. B. 2007. A complete solution of harmonics elimination problem in a multi-level inverter with unequal DC sources, J Electr Syst, 3 (4), pp. 259-271

[75] Hosseini Aghdam, M. G., Fathi, S. H., Gharehpetian, G. B. 2007. Elimination of harmonics in a multi-level inverter with unequal DC sources using the homotopy algorithm. In: Proceedings of the IEEE int. symp. ind. electron.; p. 578–83.

[76] Swift, F., Kamberis, A. 1993. A new Walsh domain technique of harmonic elimination and voltage control in pulse width modulated inverters, IEEE Trans Power Electron, 8 (2), pp. 170-185.

[77] Son, G. T., et al. 2014. Improved PD-PWM for minimizing harmonics of multilevel converter using gradient optimization. In: Proceedings of the PES

general meeting|conference & exposition. National Harbor, MD, USA; October.

[78] Chiasson, J., Tolbert, L., McKenzie, K., Du, Z. 2003. Control of a multilevel converter using resultant theory, IEEE Trans Control Syst Technol, 63 (3–5), pp. 197-208.

[79] Chiasson, J. N., Tolbert, L. M., Mckenzie, K. J., Du, Z. 2004. A complete solution to the harmonic elimination problem, IEEE Trans Power Electron, 19 (2), pp. 491-499.

[80] Imarazene, K., Chekireb, H., Berkouk, E. M. 2016. Selective harmonics elimination PWM with self- balancing DC-link in photovoltaic 7-level inverter, Turk J Electr Eng Comput Sci, 24, pp. 3999-4015.

[81] Zheng, C. F., Zhang, B. 2005. Application of Wu method to harmonic elimination techniques, Proc Chin Soc Elect Electron Eng, pp. 40-45.

[82] Yang, K., Zhang, Q., Yuan, R., Yu, W., Wang, J. 2015. Selective harmonic elimination With Groebner bases and symmetric polynomials, IEEE Trans Power Electron, 31 (4), pp. 689-694.

[83] Chiasson, J. N., Tolbert, L. M., McKenzie, K. J., Du, Z. 2005. Elimination of harmonics in a multilevel converter using the theory of symmetric polynomials and resultants, IEEE Trans Control Syst Technol, 13 (2), pp. 216-223.

[84] Chiasson, J. N., Tolbert, L. M., Du, Z., McKenzie, K. J. 2005, The use of power sums to solve the harmonic elimination equations for multilevel converters, EPE J, 15 (1), pp. 19-27.

[85] Ni, J., Wu, L., Fan, X., Yang, S. X. 2016. Bioinspired intelligent algorithm and its applications for mobile robot control: a survey, Comput Intell Neurosci.

[86] Ab Wahab, M. N., Nefti-Meziani, S., Atyabi, A. 2015. A comprehensive review of swarm optimization algorithms, PLoS One, 10 (5), pp. 1-36.

[87] Kar, A. K. 2016. Bio inspired computing – a review of algorithms and scope of applications, Expert Syst Appl, 59, pp. 20-32.

[88] Fister Jr. I., Yang, X. S., Fister, I., Brest, J., Fister, D. 2013. A brief review of nature-inspired algorithms for optimization, Neural and Evolutionary Computing, 80 (3), pp. 1-7.

[89] Yang, X. S. 2014. Swarm intelligence based algorithms: a critical analysis, Evol Intell, 7 (1), pp. 17-28.

[90] Sundareswaran, K., Jayant, K., Shanavas, T. N. 2007. Inverter harmonic elimination through a colony of continuously exploring ants, IEEE Trans Ind Electron, 54 (5), pp. 2558-2565.

[91] Lou, H., Mao, C., Wang, D., Lu, J. 2007. PWM optimisation for three-level voltage inverter based on clonal selection algorithm, IET Electr Power Appl, 1 (5), pp. 643-656.

[92] Sudhakar Babu, T., Priya, K., Maheswaran, D., Sathish Kumar, K., Rajasekar, N. 2015. Selective voltage harmonic elimination in PWM inverter using bacterial foraging algorithm, Swarm Evol Comput, 20, pp. 74-81.

[93] Salehi, R., Vahidi, B., Farokhnia, N., Abedi, M. 2010. Harmonic elimination and optimization of stepped voltage of multilevel inverter by bacterial foraging algorithm, J Electr Eng Technol, 5 (4), pp. 545-551.

[94] Lou, H., Mao, C., Wang, D., Lu, J., Wang, L. 2014. Fundamental modulation strategy with selective harmonic elimination for multilevel inverters, IET Power Electron, 7 (8), pp. 2173-2181.

[95] Wells, J. R., Geng, X., Chapman, P. L., Krein, P. T., Nee, B. M. 2007. Modulation-based harmonic elimination, IEEE Trans Power Electron, 22 (1), pp. 336-340.

[96] Ahmadi, D., Zou, K., Li, C., Huang, Y., Wang, J. 2011. A universal selective harmonic elimination method for high-power inverters, IEEE Trans Power Electron, 26 (10), pp. 2743-2752.

[97] Wang, J., Ahmadi, D. 2010. A precise and practical harmonic elimination method for multilevel inverters, IEEE Trans Ind Appl, 46 (2), pp. 857-865.

[98] Ahmadi, D., Wang, J. 2009. Study of a precise and practical harmonic elimination method for multilevel inverters. In: Proceedings of the applied power electronics conference and exposition, 2009 APEC.

[99] Al-Othman, A. K., Nabil A. Ahmed, M. E., AlSharidah, H., AlMekhaizim, A. 2013. A hybrid real coded genetic algorithm – Pattern search approach for selective harmonic elimination of PWM AC/AC voltage controller, International Journal of Electrical Power \& Energy Systems, vol. 44, no. 1, pp. 123-133.

[100] Haghdar K., Shayanfar, H. A. 2018. Selective Harmonic Elimination With Optimal DC Sources in Multilevel Inverters Using Generalized Pattern Search, IEEE Transactions on Industrial Informatics, vol. 14, no. 7, pp. 3124-3131.

[101] Haghdar, K. 2020. Optimal DC Source Influence on Selective Harmonic Elimination in Multilevel Inverters Using Teaching–Learning-Based Optimization, IEEE Transactions on Industrial Electronics, vol. 67, no. 2, pp. 942-949.

[102] Sundareswaran, K., Jayant K., Shanavas, T. N. 2007. Inverter Harmonic Elimination Through a Colony of Continuously Exploring Ants, IEEE Transactions on Industrial Electronics, vol. 54, no. 5, pp. 2558-2565.

[103] Ganesan, K., Barathi, K., Chandrasekar, P., Balaji, D. 2015. Selective Harmonic Elimination of Cascaded Multilevel Inverter Using BAT Algorithm, Procedia Technology, vol. 21, pp. 651-657.

[104] Dahidah, M. S. A., Rao, M. V. C. 2007. A Hybrid Genetic Algorithm for Selective Harmonic Elimination PWM AC/AC Converter, Control. Electr. Eng., vol. 89, pp. 285–291.

[105] Sen, P., Bana, P. R., Panda, K. P. 2019. Firefly Assisted Genetic Algorithm for Selective Harmonic Elimination in PV Interfacing Reduced Switch Multilevel Inverter, International Journal of Renewable Energy Research, 9(1).

[106] Sudhakar Babu, T., Priya, K., Maheswaran, D., Sathish Kumar, K., Rajasekar, N. 2015. Selective voltage harmonic elimination in PWM inverter using bacterial foraging algorithm, Swarm and Evolutionary Computation, vol. 20, pp. 74-81.

[107] Rajaram, R., Palanisamy Sudha Ramasamy Prabhu Ramanathan, K. 2014. Selective Harmonic Elimination in PWM Inverter Using Fire Fly and Fire Works Algorithm, International Journal of Innovative Research in Advanced Engineering (IJIRAE), 1(8).

[108] Dzung, P. Q., Tien, N. T., Dinh Tuyen, N., Lee, H. 2015. Selective harmonic elimination for cascaded multilevel inverters using Grey Wolf Optimizer algorithm, 9th International Conference on Power Electronics and ECCE Asia (ICPE-ECCE Asia), Seoul, pp. 2776-2781.

[109] Nalcaci, G., Ermis, M. 2019. Effect of Grey Wolf Optimization on THD of 3-Phase Voltage Source Inverter with Selective Harmonic Elimination Base, 4th International Conference on Power Electronics and their Applications (ICPEA), Elazig, Turkey, pp. 1-5.

[110] Nalcaci, G., Ermis, M. 2018. Selective Harmonic Elimination for Three-Phase Voltage Source Inverters Using Whale Optimizer Algorithm, 5th International Conference on Electrical and Electronics Engineering (ICEEE), Istanbul, Turkey, pp. 1-6.

[111] Franceschini, G., Lorenzani, E., Cavatorta, M., Bellini, A. 2008. 3boost: A High-Power Three-Phase Step-Up Full-Bridge Converter for Automotive Applications, in IEEE Transactions on Industrial Electronics, vol. 55, no. 1, pp. 173-183, doi: 10.1109/TIE.2007.905930.

[112] Kim, J., Kwon J., Kwon, B. 2018. High-Efficiency Two-Stage Three-Level Grid-Connected Photovoltaic Inverter, in IEEE Transactions on Industrial Electronics, vol. 65, no. 3, pp. 2368-2377, doi: 10.1109/TIE.2017.2740835.

[113] https://www.skeletontech.com/train-and-rail-industry

[114] Kalla-Bishop, P. M. 1972. Future Railways and Guided Transport, IPC Transport Press Ltd., pp. 8-33

[115] A train ride through history. SWI swissinfo.ch.

[116]A nation of railway enthusiasts: a history of the Swiss railways. House of Switzerland.

[117]Indian Railways sets new benchmark! Runs 1st Double-stack container train in high rise OHE electrified sections. 12 June 2020.

[118]Spotlight on double-stack container movement. @businessline. Retrieved 1 July 2020.

[119]Aerodynamic Effects Caused by Trains Entering Tunnels. ResearchGate. Retrieved 1 July 2020.

[120]Railway Handbook 2015. International Energy Agency. p. 18. Retrieved 4 August 2017.

[121]EN 50163: Railway applications. Supply voltages of traction systems 2007.

[122]IEC 60850: Railway applications – Supply voltages of traction systems, 3rd edition 2007.

[123]Leandes, P., Ostlund, S. 1989. A concept for an HVDC traction system in International conference on main line railway electrification, Hessington, England.

[124]Glomez-Exposito, A., Mauricio, J. M., Maza-Ortega, J. M. 2014. VSC-based MVDC Railway Electrification System, IEEE transactions on power delivery, v.29, no.1..

[125]Patrobers, S., Davidson, I. E. 2021. MVDC Railway Traction Power Systems; State-of-the Art, Opportunities, and Challenges, Energies, MDPI. 14 (14). doi:10.3390/en14144156. ISSN 1996-1073.

[126]Güvengir, U. 2014. Online application of SHEM to grid-connected inverters with variable DC link voltage by particle swarm optimization.

[127]Majid, K. I. 1974. Optimum design of structures. Newness-Butterworth, UK.

[128]Kirsch, U. 1993. Structural optimization: fundamentals and applications. Springer, Berlin, Heidelberg

[129]Gonzalez, T. F. 2007. Handbook of approximation algorithms and metaheuristics, Computer and information science series. Chapman & Hall/CRC, Boca Raton, FL

[130]Yang, X. S. 2010 Nature-inspired metaheuristic algorithms, 2nd edn. Luniver Press, UK

[131]Glover, F., Kochenberger, G. A. 2003, Handbook of metaheuristics. Kluwer Academic Publishers, Dordrecht.

[132]Kennedy J., Eberhart, R. 1995. Particle swarm optimization, Proceedings of ICNN'95 - International Conference on Neural Networks, pp. 1942-1948 vol.4, doi: 10.1109/ICNN.1995.488968.

[133]Eberhart, Y. S., "Particle swarm optimization: developments, applications and resources," Proceedings of the 2001 Congress on Evolutionary Computation (IEEE Cat. No.01TH8546), 2001, pp. 81-86 vol. 1, doi: 10.1109/CEC.2001.934374.

[134]Mirjalili, S., Mirjalili, S. M., Lewis, A. 2014. Grey Wolf Optimizer, Advances in Engineering Software, 69, 2014, 46-61.

[135]Mirjalili, S., Lewis, A., 2016, The Whale Optimization Algorithm, Advances in Engineering Software, 95, 51-67.

[136]Heidari, A. A., Mirjalili, S., Faris, H., Aljarah, I., Mafarja, M., Chen, H. 2019. Harris hawks optimization: Algorithm and applications, Future Generation Computer Systems, 97, 849-872.

[137]"Fact Book Glossary - Mode of Service Definitions". American Public Transportation Association. 2015.

[138]"National Transit Database Glossary". U.S. Department of Transportation Federal Transit Administration. 2013.

[139]"What is light rail?". Public transport A-Z. International Association of Public Transport. 2008.

[140]"This Is Light Rail Transit". Transportation Research Board. pp. 7–9. 2018.

[141]"What is Light Rail?". Light Rail Transit Association(LRTA). 2016.

[142]Wang, J., S., Li, S., X. 2019. An Improved Grey Wolf Optimizer Based on Differential Evolution and Elimination Mechanism, Sci Rep 9, 7181.

[143]Railway applications - Supply voltages of traction systems, EN 50163:2004/A1:2007.

[144]Mohan, N. 2002. Power Electronics: Converters, Applications and Design, John Wiley and Sons.

APPENDICES

A. PSO CUDA CODE

```
#include <iostream>
#include <cstdio>
#include <cstdlib>
#include <cmath>
#include <ctime>
#include <cuda_runtime.h>
#include <device_launch_parameters.h>
#include <curand.h>
#include <curand_kernel.h>

using namespace std;

#define dims 7                    //number of dimensions in fitness function
#define inf 9999.99f               //infinity
#define x_min 0.0f                //minimum x
#define x_max 90.0f                //maximum x
#define max_iters 100              //number of iterations
#define max_particles 30          //number of particles
#define chi 0.72984f               //chi (constriction factor)
#define pi 3.14159265f             //value of pi

#define cudaCheckError()\
{\
    cudaError_t e = cudaGetLastError();\
    if(e != cudaSuccess)\
    {\
        printf("CUDA failure: %s:%d: %s\n", __FILE__, __LINE__,
cudaGetErrorString(e));\
        exit(EXIT_FAILURE);\
    }\
}

__global__
void Initialize(float* pos, float* velocity, float* p_best_y, int*
l_best_index, int* best_index, curandState* states)
{
    int index = blockDim.x * blockIdx.x + threadIdx.x;
    int t_index = threadIdx.x;

    pos[index] = x_max * (2.0f * pos[index] - 1.0f);

    velocity[index] = 0.5f * velocity[index] * (x_max - x_min) / 2.0f;

    if (t_index == 0)
    {
        p_best_y[blockIdx.x] = inf;
        l_best_index[blockIdx.x] = blockIdx.x;
        best_index[blockIdx.x] = blockIdx.x;
    }
```

```
    curand_init(index, index, 0, &states[index]);
}
__global__
void Iterate(float* pos, float* velocity, float* p_best_x, float* p_best_y,
int* l_best_index, curandState* states)
{
    int index = blockDim.x * blockIdx.x + threadIdx.x;
    float personal_best;
    int local_best;
    curandState local_state = states[index];
    float v_max = 0.5f * (x_max - x_min) / 2.0f;
    float c1 = 2.05f, c2 = 2.05f;
    float r1, r2;
    int M=0.7;
    int left = (max_particles + index - 1) % max_particles;
    int right = (1 + index) % max_particles;
    float fitness = 0.0f;
    float fitness1 = 0.0f;

    for (int i = 0; i < dims; i++)
        fitness1 = pow(-1,i)*cos(pos[index + i])-M;
    for (int i = 0; i < dims; i++)
        fitness += pow(-1, i) * cos((2 * i + 1) * pos[index + i]);
    fitness = fitness + fitness1;

    if (p_best_y[index] > fitness)
    {
        p_best_y[index] = fitness;
        for (int i = 0; i < dims; i++)
            p_best_x[index * dims + i] = pos[index * dims + i];
    }
    personal_best = p_best_y[index];

    if (p_best_y[left] < personal_best)
        l_best_index[index] = left;
    if (p_best_y[right] < personal_best)
        l_best_index[index] = right;
    local_best = l_best_index[index];

    for (int i = 0; i < dims; i++)
    {
        int id = index * dims + i;
        r1 = curand_uniform(&local_state);
        r2 = curand_uniform(&local_state);

        velocity[id] = chi * (velocity[id] + (c1 * r1 * (p_best_x[id] -
pos[id])) + (c2 * r2 * (p_best_x[local_best] - pos[id])));

        if (velocity[id] > v_max)
            velocity[id] = v_max;
        if (velocity[id] < -v_max)
            velocity[id] = -v_max;
        pos[id] = pos[id] + velocity[id];

        if (pos[id] > x_max)
```

```
            pos[id] = x_max;
        if (pos[id] < -x_max)
            pos[id] = -x_max;
    }

    states[index] = local_state;
}

__global__
void Reduce(float* p_best_x, float* p_best_y, int* best_index, int step)
{
    int index = blockDim.x * blockIdx.x + threadIdx.x;
    int tx = threadIdx.x;

    __shared__ float stage[512];
    __shared__ int best[512];

    best[tx] = best_index[index];
    stage[tx] = p_best_y[index];
    __syncthreads();

    for (unsigned int s = blockDim.x / 2; s > 0; s >>= 1)
    {
        if (tx < s)
        {
            if (stage[tx] > stage[tx + s])
            {
                stage[tx] = stage[tx + s];
                best[tx] = best[tx + s];
            }
        }
        __syncthreads();
    }

    if (tx == 0)
    {
        p_best_y[blockIdx.x] = stage[0];
        best_index[blockIdx.x] = best[0];
    }

    if (step == 2)
    {
        for (int i = 0; i < dims; i++)
            p_best_x[i] = p_best_x[best[0] * dims + i];
    }
}

int main()
{
    cout << endl;

    float* g_best;
    float* g_best_pos;
    float* pos, * velocity;
    float* p_best_x, * p_best_y;
    int* l_best_index, * best_index;
```

```
    float ms = 0;
    g_best = new float;
    g_best_pos = new float[dims];

    curandState* states;

    cudaEvent_t start, stop;
    cudaEventCreate(&start);
    cudaEventCreate(&stop);

    cudaEventRecord(start);
    cudaMalloc((void**)&pos, max_particles * dims * sizeof(float));
    cudaCheckError();

    cudaMalloc((void**)&velocity, max_particles * dims * sizeof(float));
    cudaCheckError();
    cudaMalloc((void**)&p_best_x, max_particles * dims * sizeof(float));
    cudaCheckError();

    cudaMalloc((void**)&p_best_y, max_particles * sizeof(float));
    cudaCheckError();

    cudaMalloc((void**)&l_best_index, max_particles * sizeof(int));
    cudaCheckError();

    cudaMalloc((void**)&best_index, max_particles * sizeof(int));
    cudaCheckError();

    cudaMalloc((void**)&states, max_particles * dims * sizeof(curandState));
    cudaCheckError();

    curandGenerator_t gen;
    curandCreateGenerator(&gen, CURAND_RNG_PSEUDO_DEFAULT);
    curandSetPseudoRandomGeneratorSeed(gen, time(NULL));

    curandGenerateUniform(gen, pos, max_particles * dims);
    cudaCheckError();

    curandGenerateUniform(gen, velocity, max_particles * dims);
    cudaCheckError();

    Initialize << <max_particles, dims >> > (pos, velocity, p_best_y,
l_best_index, best_index, states);
    cudaCheckError();

    cudaEventElapsedTime(&ms, start, stop);
    cout << "Initialization: time taken: " << ms << " millisec" << endl;

    cout << endl

    for (int i = 0; i < max_iters; i++)
        Iterate << <max_particles / 32, 32 >> > (pos, velocity, p_best_x,
p_best_y, l_best_index, states);
```

```
    cout << "Iterations: time taken: " << ms << " millisec" << endl;

    cout << endl;

    Reduce << <max_particles / 32, 32 >> > (p_best_x, p_best_y, best_index,
1);
    cudaCheckError();

    Reduce << <1, max_particles / 32 >> > (p_best_x, p_best_y, best_index,
2);
    cudaCheckError();

    cudaMemcpy((void*)g_best, (void*)p_best_y, sizeof(float),
cudaMemcpyDeviceToHost);
    cudaCheckError();

    cudaMemcpy((void*)g_best_pos, (void*)p_best_x, dims * sizeof(float),
cudaMemcpyDeviceToHost);
    cudaCheckError();

    cudaEventRecord(stop);
    cudaEventSynchronize(stop);

    cout << "Reduce: time taken: " << ms << " millisec" << endl;

    cout << endl;

    //Print results
    cout << "Global minimum is: " << *g_best << endl;
    cout << "At:" << endl;

    for (int i = 0; i < dims; i++)
        cout << "x[" << i << "] = " << g_best_pos[i] << endl;

    cudaFree(pos);
    cudaFree(velocity);
    cudaFree(p_best_x);
    cudaFree(p_best_y);
    cudaFree(l_best_index);
    cudaFree(best_index);
    cudaFree(states);
    cudaCheckError();

    curandDestroyGenerator(gen);
    cudaCheckError();

    cout << endl;

    return 0;
}
```

B. GWO CUDA CODE OF ITERATION FUNCTION

```
__global__
void Iterate(float* pos, float* searchAgents_no, float* dim, int* lb, int*
ub, curandState* states)
{
    int index = blockDim.x * blockIdx.x + threadIdx.x;
    int Alpha_score, Beta_score, Delta_score;
    curandState local_state = states[index];

    float Max_iter;
    float l = 0.0f;
    float Alpha_pos, Beta_pos, Delta_pos;
    float X1, X2, X3;
    float A1, A2, A3;
    float D_alpha, D_beta, D_delta;
    int M=0.7;
    float fitness = 0.0f;
    float fitness1 = 0.0f;

    for (int i = 0; i < dims; i++)
        fitness1 = pow(-1,i)*cos(pos[index + i])-M;
    for (int i = 0; i < dims; i++)
        fitness += pow(-1, i) * cos((2 * i + 1) * pos[index + i]);
    fitness = fitness + fitness1;
    for (int i = 0; i < dims; i++)
    if (fitness<Alpha_score)
    {
        Alpha_score = fitness;
        Alpha_pos = pos[index * dims + i];
    }
       if (fitness>Alpha_score && fitness<Beta_score)
    {
        Beta_score = fitness;
        Beta_pos = pos[index * dims + i];
    }
    if (fitness>Alpha_score && fitness>Beta_score && fitness<Delta_score)
    {
        Delta_score = fitness;
        Delta_pos = pos[index * dims + i];
    }
        a=2-l*(2/Max_iter);

    for (int i = 0; i < sizeof(pos[index,1]); i++)
    {
       for (int j = 0; j < sizeof(pos[index,2]); j++)
       {
        int id = index * dims + i;
        r1 = curand_uniform(&local_state);
        r2 = curand_uniform(&local_state);
        A1 = 2*a*r1-a;
        C1 = 2*r2;
        D_alpha = abs(C1*Alpha_pos(j)- pos([index * dims + i],j));
```

```
            X1 = Alpha_pos(j)-A1*D_alpha;
            r1 = curand_uniform(&local_state);
            r2 = curand_uniform(&local_state);
            A2 = 2*a*r1-a;
            C2 = 2*r2;
            D_beta = abs(C2*Beta_pos(j)- pos([index * dims + i],j));
            X3 = Delta_pos(j)-A3*D_delta;
            r1 = curand_uniform(&local_state);
            r2 = curand_uniform(&local_state);
            A3 = 2*a*r1-a;
            C3 = 2*r2;
            D_delta = abs(C3*Delta_pos(j)- pos([index * dims + i],j));
            X3 = Delta_pos(j)-A3*D_Delta;
        }

        states[index] = local_state;
    }
```

C. WOA CUDA CODE OF ITERATION FUNCTION

```
__global__
void Iterate(float* pos, float* searchAgents_no, float* dim, int* lb, int*
ub, curandState* states)
{
    int index = blockDim.x * blockIdx.x + threadIdx.x;
    curandState local_state = states[index];

    float Max_iter;
    float l = 0.0f;
    float Leader_score;
    float Leader_pos;
    float a, a2;
    float A1, C, b, p;
    float rand_leader_index, X_rand, D_X_rand, D_Leader, distance2Leader;
    int M=0.7;
    float fitness = 0.0f;
    float fitness1 = 0.0f;

    for (int i = 0; i < dims; i++)
        fitness1 = pow(-1,i)*cos(pos[index + i])-M;
    for (int i = 0; i < dims; i++)
        fitness += pow(-1, i) * cos((2 * i + 1) * pos[index + i]);
    fitness = fitness + fitness1;
    if (fitness<Leader_score)
    {
        Leader_score = fitness;
        Leader_pos = pos[index * dims + i];
    }
        a=2-l*(2/Max_iter);
        a2=-1+t*((-1)/Max_iter)

     for (int i = 0; i < sizeof(pos[index,1]); i++)
    {
        int id = index * dims + i;
        r1 = curand_uniform(&local_state);
        r2 = curand_uniform(&local_state);
        A1 = 2*a*r1-a;
        C = 2*r2;
        b =1;
        p = pos([index],2);
        if (p<0.5 && abs(A)>=1)
    {
        rand_leader_index = floor(searchAgents_no*curand_init(index, index,
       0, &states[index])+1);
        X_rand = pos([rand_leader_index],:);
        D_X_rand = abs(C*X_rand(j)- pos([index * dims + i],j);
        pos([index * dims + i],j) = X_rand(j)-A*D_X_Rand;
    }

        elseif (abs(A)<1)
    {
```

```
        D_Leader = abs(C*Leader_pos(j)- pos([index * dims + i],j);
        pos([index * dims + i],j) = Leader_pos(j)-A*D_Leader;
    }
           elseif (p>=0.5)
    {
        distance2Leader = abs(C*Leader_pos(j)- pos([index * dims + i],j);
        pos([index * dims + i],j) =
       distance2Leader*exp(b.l).*cos(l.*2*pi)+Leader_pos(j);
    }

    states[index] = local_state;
}
```

D. HHO CUDA CODE OF ITERATION FUNCTION

```
void Levy(float*d)
{
       float beta = 1.5f;
       float sigma, u, v, step, o;
       sigma =
(gamma(1+beta)*sin(pi*beta/2)/(gamma((1+beta)/2)*beta*2^((beta-
1)/2)))^(1/beta);
       u = curand_init(index, index, 0, &states[index])*sigma;
       v = curand_init(index, index, 0, &states[index]);
       step=u./abs(v).^(1/beta);
       o = step;
}

__global__
void Iterate(float* pos, float* searchAgents_no, float* dim, int* lb, int*
ub, curandState* states)
{
    int index = blockDim.x * blockIdx.x + threadIdx.x;
    int Alpha_score, Beta_score, Delta_score;
    curandState local_state = states[index];

    float Max_iter;
    float l = 0.0f;
    float Leader_score;
    float Leader_pos;
    float a, a2;
    float A1, C, b, p;
    float rand_leader_index, X_rand, D_X_rand, D_Leader, distance2Leader;
    int M=0.7;
    float fitness = 0.0f;
    float fitness1 = 0.0f;

    for (int i = 0; i < dims; i++)
        fitness1 = pow(-1,i)*cos(pos[index + i])-M;
    for (int i = 0; i < dims; i++)
        fitness += pow(-1, i) * cos((2 * i + 1) * pos[index + i]);
    fitness = fitness + fitness1;
    for (int i = 0; i < X; i++)
        FU = X(i,:)>ub;  FL = X(i,:)<lb;
X(i,:)=(X(i,:).*(~(FU+FL)))+ub.*FU+lb.*FL;
     if (fitness<Rabbit_Energy)
    {
        Rabbit_Energy = fitness;
        Rabbit_Location = pos[index * dims + i];
    }
        E1 = 2*(1-t/T);
    for (int i = 0; i < sizeof(X); i++)
        E0 = 2*curand_init(index, index, 0, &states[index])-1;
        Escaping_Energy = E1*(E0);
    if (abs(Escaping_Energy)>=1)
    {
```

```
 q = curand_init(index, index, 0, &states[index]);
 rand_Hawk_index = floor(N* curand_init(index, index, 0,
&states[index])+1);
 X_rand = X(rand_Hawk_index,:);
 if (q<0.5)
{
     X(i,:) = X_rand-curand_init(index, index, 0,
&states[index])*abs(X_rand-2* curand_init(index, index, 0,
&states[index])* X(i,:));
}
elseif (q>=0.5)
{
     X(i,:) = (Rabbit_Location(1,:)-mean(X))-curand_init(index, index,
0, &states[index])*((ub-lb)* curand_init(index, index, 0,
&states[index])+lb);
}
elseif (abs(Escaping_Energy<1))
{
     r= curand_init(index, index, 0, &states[index])+lb);
     if (r>=0.5 && abs(Escaping_Energy))
     {
     X(i,:) = Rabbit_Location-Escaping_Energy*abs(Rabbit_Location-
X(i,:))
     }
     if (r>=0.5 && abs(Escaping_Energy)>=0.5)
     {
     Jump_strength=2*(1-curand_init(index, index, 0, &states[index]));
     X(i,:) = Rabbit_Location-X(i,:)-
Escaping_Energy*abs(Jump_strength*Rabbit_Location-X(i,:));
     }
     if (r<0.5 && abs(Escaping_Energy)>=0.5)
     {
     Jump_strength=2*(1-curand_init(index, index, 0, &states[index]));
     X1 = Rabbit_Location-
Escaping_Energy*abs(Jump_strength*Rabbit_Location-X(i,:));
     }
     if (fobj(X1)<fobj(X(i,:)))
     {
     X(i,:)=X1;
     }
     else
     {
     X2 = Rabbit_Location-
Escaping_Energy*abs(Jump_strength*Rabbit_Location-X(i,:)+
curand_init(index, index, 0, &states[index])*Levy(dim);
     }
     if(fobj(X2)<fobj(X(i,:)))
     {
     X(i,:) = X2;
     }
     if(r<0.5 && abs(Escaping_Energy)<0.5)
     {
     Jump_strength = 2*(1- curand_init(index, index, 0,
&states[index]));
     X1 = Rabbit_Location-
Escaping_Energy*abs(Jump_strength*Rabbit_Location-mean(X));
```

```
            }
            if(fobj(X1)<fobj(X(i,:)))
            {
            X(i,:) = X1;
            }
            else
            {
            X2 = Rabbit_Location-
        Escaping_Energy*abs(Jump_strength*Rabbit_Location-mean(X))+
        curand_init(index, index, 0, &states[index])*Levy(dim);
            }
            if(fobj(X2)<fobj(X(i,:)))
            {
            X(i,:) = X2;
            }
        }
    }
    states[index] = local_state;
}
```

CURRICULUM VITAE

PERSONAL INFORMATION

Surname, Name: Nalçacı, Gamze
Nationality: Turkish (TC)
Date and Place of Birth: 23 January 1989, Çorum
Marital Status: Single
Phone: +90 312 210 45 89
Fax: +90 312 210 23 04
email: gnalcaci@metu.edu.tr

EDUCATION

Degree	Institution	Year of Graduation
BS	Selcuk University Electrical and Electronics Engineering	2011
High School	Corum Anatolian High School	2007

WORK EXPERIENCE

Year	Place	Enrollment
2021-Present	Necmettin Erbakan University Dept. of Electrical and Electronics Eng.	Research Assistant
2012-2020	METU Dept. Of Electrical and Electronics Eng.	Research Assistant
2011-2012	Necmettin Erbakan University Dept. of Electrical and Electronics Eng.	Research Assistant
2010 July	TAI-TUSAŞ	Intern Eng. Student
2009 July	Aselsan	Intern Eng. Student

FOREIGN LANGUAGES

Advanced English, Beginner German

PUBLICATIONS

Journals

1. Nalcaci G., Özmen A., Weber G.W., "Long-term load forecasting: models based on MARS, ANN and LR methods", Central European Journal Operations Research 27, 1033–1049, 2019.
2. Nalcaci G., Yildirim D., Cadirci I., Ermis M., "Selective Harmonic Elimination for Variable Frequency Traction Motor Drives Using Harris Hawks Optimization", IEEE Transactions on Industry Applications, *In Review.*

Proceedings

1. Nalcaci G., Yildirim D., Ermis, M., "Selective Harmonic Elimination for Light-Rail Transportation Motor Drives using Harris Hawks Algorithm", IEEE International Conference on Environment and Electrical Engineering (EEEIC) and IEEE Industrial and Commercial Power Systems Europe (EEEIC / I&CPS Europe), Madrid, Spain, 2020, pp. 1-6.
2. Nalcaci G., Nalcaci G., "Modeling and Implementation of an Adaptive Facade Design for Energy Efficiently Buildings Based Biomimicry", 8th International Conference on Smart Grid (icSmartGrid), Paris, France, 2020, pp. 140-145.
3. Shabani R., Nalcaci G., Leblebicioglu M. K., Ermis M., "Cost Function Determination for a WIG in Predefined Path and Height Using Conjugate Gradient Method", 2020 IEEE 6th International Conference on Control Science and Systems Engineering (ICCSSE), Beijing, China, 2020, pp. 1-6.
4. Nalcaci G., Ermis M., "Effect of Grey Wolf Optimization on THD of 3-Phase Voltage Source Inverter with Selective Harmonic Elimination Base", 4th International Conference on Power Electronics and their Applications (ICPEA), Elazig, Turkey, 2019, pp. 1-5.
5. Nalcaci G., Ermis M., "Selective Harmonic Elimination for Three-Phase Voltage Source Inverters Using Whale Optimizer Algorithm", 5th International Conference on Electrical and Electronics Engineering, Istanbul (ICEEE), Turkey, 2018, pp. 1-6.