ELECTRIC FENCE AGAINST LOCUSTS

TOOBA QAISER
ERZA ZAHOOR
ESHA JAVED
ANIQA AFZAL

DEPARTMENT OF ELECTRICAL ENGINEERING

LAHORE COLLEGE FOR WOMEN UNIVERSITY,

LAHORE

2023

BS
THESIS

ELECTRIC FENCE AGAINST LPCUST

2023

ELECTRIC FENCE AGAINST LOCUSTS

A THESIS SUBMITTED TO LAHORE COLLEGE FOR WOMEN UNIVERSITY IN PARTIAL FULFILMENT OF THE REQUIREMENTS FOR THE DEGREE OF BS IN ELECTRICAL ENGINEERING

By
TOOBA QAISER
ESHA JAVED
ERZA ZAHOOR
ANIQA AFZAL

DEPARTMENT OF ELECTRICAL ENGINEERING

LAHORE COLLEGE FOR WOMEN UNIVERSITY, LAHORE
2023

CERTIFICATE

This is to certify that the research work described in this thesis submitted by **MS. TOOBA QAISER, MS. ANIQA AFZAL, MS. ERZA ZAHOOR and MS. ESHA JAVED** to Department of Electrical Engineering, Lahore College for Women University has been carried out under my direct supervision. I have personally gone through the raw data and certify the correctness and authenticity of all results reported herein. I further certify that thesis data have not been used in part or full, in a manuscript already submitted or in the process of submission in partial fulfillment of the award of any other degree from any other institution or home or abroad. I also certify that the enclosed manuscript has been prepared under my supervision and I endorse its evaluation for the award of BS/MS/PhD degree through the official procedure of the University.

_____
Dr Faiza Iftikhar
Supervisor
Date:

_____
Name
Co-Supervisor
Date:

Verified By

_____
Name
Chairperson
Department of _____
Stamp

_____
Controller of Examination
Stamp
Date: _____

DEDICATION

We dedicate this book to ALLAH Almighty, thank you for the guidance, strength, power of the mind, protection and skills and for giving us a healthy life. Then dedicated to our beloved Holy Prophet Muhammad (PBUH). To our beloved parents and teachers, who have been our source of inspiration and gave us strength when we thought of giving up. To our brothers, sisters, relatives, friends and classmates who shared their words of advice and encouragement to finish this study.

ACKNOWLEDGMENTS

All praises to Allah, who has given us strength, knowledge and his blessings in finishing this work. Many prayers and peace be upon the last and final Prophet Muhammad (S.A.W)

This work has been done at the Department of Electrical Engineering, Lahore College for Women University, Pakistan. This thesis would not be possible without the help of many people. Firstly, we would like to thank Chairperson, Dr Sadia, for her support and suggestions. Especially we would like to thank and be profoundly indebted to our research project supervisor, Dr Faiza Iftikhar, for giving us the great opportunity to do research with her. She enthused us to choose this exceptional topic, we enjoyed every bit of it during our research period when we were doing our project. Without her help and valuable guidance, we couldn't even start it.

And finally, the endless gratitude towards our parents, siblings and friends that cannot be expressed in words.

CONTENTS

Title **Page No.**

List of Table **i**

List of Figures **ii**

List of Abbreviations **iii**

Abstract **iv**

Chapter 1 : Introduction **12**

1.1 . Introduction 13
1.1.1. 2019–2022 locust infestation 13
1.1.2. Causes 13
1.1.3. Most Affected Countries 14
1.1.4. Effect of Climate 15
1.2 . Motivation 15
1.3 . Problem Statement 16
1.4 . Electric Fence 16
1.4.1. Design and Function 17
1.4.2. Fencing Materials 17
1.5. Applications 17
1.5.1. Use in Agriculture 17
1.5.2. Buried electric fences 18

Chapter 2: Literature Review

2.1. Previous Work 20
2.2. Efforts of Pakistan 21
2.2.1. During COVID-19 22
2.2.2. Aid from JICA 22
2.2.3. Enhancing locust control and monitoring capabilities to prepare for future attacks 22
2.3. Efforts by other countries 23
2.4.1. Help from World Bank 23

Chapter 3: Electric Fence for Locusts

3.1. Proposed Solution 25
3.1.1. Working of Voice Module 25
3.1.2. Activation of Electric Field 26
3.1.3. Working of the Whole Prototype 26
3.1.4. Expected Result 26
3.2. Hardware Implementation 27
3.2.1. Mathematical Values 27
3.3. Hardware Components 28
3.3.1. ZVS driver 28
3.3.2. ZVS (Zero Voltage Switching) flyback transformer driver 29
3.3.3. Working 29
3.4. Voice Recognition Module 30
3.4.1. Feature 30
3.4.2. Operating Voltages 31
3.5. Relay 31
3.5.1. Working 31
3.5.2. Advantages of using Relay 33
3.6. Arduino 33
3.6.1. Working 34
3.7. Software Implementation 35
3.8. Training of voice commands 37
3.8.1. Add Voice Command 38

Chapter 4: Results

4.1. Reduction in Locust Incursions 40
4.2. Behavioral Responses 40
4.3. Fence Design and Configuration 40
4.4. Crop Damage Reduction 41

Chapter 5: Discussion

5.1. Discussion 43
5.1.1. Electric Fence System 43
5.1.2. Comparative Analysis 43
5.2. Advantages and Impacts 44
5.2.1. Eco-friendly Solution 44
5.2.2. Cost-effective and efficient 44
5.3. Design Issues 44
5.3.1. Insufficient Voltage 44
5.3.2. Inadequate Coverage 44
5.3.3. Grounding Issues 45
5.3.4. Vegetation Interference 45
5.3.5. Equipment Malfunction 45
5.3.6. Environmental Factors 45
5.3.7. Locust Adaptation 45
5.3.8. Human Error 45

5.4. SDGs that are mapping our Project 46
5.4.1. Establish Good Health And Well-Being 46
5.4.2. Erase Hunger 46
5.4.3. Create Decent Work and Economic Growth 46
5.5. Environment, Health and Safety 46
5.6. CEP Attributes 46
5.7. CEA Attributes 47
5.8. Conclusion 47
5.9. Future Enhancement 48
5.9.1. Intelligent and adaptive systems 48
5.9.2. Integrated pest management 48
5.9.3. Remote monitoring and control 48
5.9.4. Data analytics and predictive modeling 48
5.9.5. Energy efficiency and sustainability 48
5.9.6. Public awareness and stakeholder engagement 49
5.9.7. Magnetic Field 49
Appendix
References 50

List of Figures

Figure No.	Title	Page No.
1	Flowchart	25
2	Arduino and Voice Module V3	36
3	Arduino Software	37
4	Prototype	40

Abstract

This project gives the solution to protect the farms from locusts through electric field. Locusts are creating a lot of destruction to agricultural lands, which results in famine and starvation. These locusts have ruined wheat, cotton and vegetable crops in Pakistan since the 1990s. This project acts as a safeguard for food. It consists of electrodes that are used to create an electric field which kills the locusts by giving them electric shocks. A Voice module is also used to sense the presence of locusts.

CHAPTER NO. 1

INTRODUCTION

1.1. Introduction

A fence can be defined as a structure that serves as a barrier and is usually connected by wires. Whereas, electric fence functions as a boundary, but rely on electricity to keep the insects away from farms. As we know Pakistan is an agricultural country. Our economy is predominantly based on agriculture and devastating fear to our herbaceous vegetation is locusts. Locusts are migratory grasshoppers that travel in massive horde and can grow rapidly. Furthermore, the individuals bear different behavioral states, color and body shape when the population density is high.

1.1.1. 2019–2022 locust infestation

A pest outbreak of desert locusts is threatening the supply of food in East Africa, the Arabian Peninsula, and the Indian subcontinent from 2019 to 2022. The flare-up is the most horrendously terrible in 70 years in Kenya and the most obviously awful in 25 years in Ethiopia, Somalia, and India [1].

The plague started in June 2019. By the end of 2019, there were swarms in Ethiopia, Eritrea, Somalia, Kenya, Saudi Arabia, Yemen, Egypt, Oman, Iran, India, and Pakistan. By June 2020, a separate swarm appeared in South America, affecting Paraguay and Argentina [2].

In April 2020, travel and shipping restrictions precipitated by the spread of COVID-19 began to hamper efforts to control the locusts, preventing the transport of pesticides, equipment, and personnel Only Ethiopia, Eritrea, Somalia, and Yemen had significant swarms of gregarious locusts by October 2020 [3]; the remaining population was concentrated in Kenya, Sudan, and Saudi Arabia in isolated areas. Until February 2022, when the surge was officially declared over, locust swarms continued to pose a threat to countries surrounding the southern Red Sea and the Gulf of Aden as well as their immediate neighbors [4].

1.1.2. Causes

Solitary desert locusts emerge to feed on new-growth foliage and lay eggs in newly moist soil following periods of relative drought to prevent them from drying out [5] . The sight and smell of other locusts, in addition to sensory stimulation from contact between locusts' hind legs, precipitate behavioral and morphological changes in the locusts; the beforehand green, nighttime and singular animals become bigger, foster dark and-yellow shading, and start to search out different beetles, a cycle known as gregarization [6]. These progressions bring about the development of enormous gregarizing swarms close by grasshoppers and breed plentifully, permitting them to go through fast, dramatic development. The multitudes continue to benefit from the recently bountiful vegetation, utilizing further developed swarm coordination, the consequence of bigger cerebrum

sizes, as well as expanded range, the consequence of expanded metabolic movement, bigger muscles, and longer wings, to make a trip up to 130 km daily looking for new vegetation and soggy climate, frequently pushing themselves by the breeze [7].

This particular desert locust plague began in May 2018, when Cyclone Mekunu passed over the Rub' al Khali, a vast, uninhabited desert in the southern Arabian Peninsula. The cyclone created ephemeral lakes in the space between the sand dunes, allowing locusts to breed undetected. This was made worse in October 2018 by Cyclone Luban, which started in the middle of the Arabian Sea, moved west, and pounded the same area near the Yemeni-Omani border [8, 9].

The Indian Ocean Dipole, an irregular oscillation in sea surface temperatures that occurs between the western and eastern parts of the Indian Ocean, has increased in magnitude as a result of the effects of climate change [10]. This shift has prompted an expansion in the number of typhoons in the Persian Bay, which was once home to not many twisters. It is additionally connected to bushfires in Australia [11], flooding in nations along the western Indian Sea, and dry climate in the east.

The two twisters created conditions favourable to widespread grasshopper propagation, allowing three generations of beetles to multiply over a nine-month period, resulting in an approximately 8,000-overlay increase in their number in the Middle Eastern desert.

1.1.3. Most Affected Countries

Africa: Swarms had crossed the Red Sea and the Gulf of Aden into Ethiopia and Somalia by the summer of 2019, where they continued to breed and started causing concern. If it weren't for the unusually widespread and intense autumn rains that fell on East Africa in October 2018 and were capped in December by the rare late-season cyclone Pawan, which made landfall in Somalia, this might have been the closest the locusts got. The occurrence of these things sparked yet another flurry of reproduction [11].

Kenya: Infesting 21 counties by the end of February and reaching Kenya's borders with Uganda and Tanzania, several large, immature swarms were first reported to have crossed into Kenya from Somalia on December 28. Heavy rains during the preceding short rain season (October–December) created an environment conducive to locust breeding, and over the next two months, the swarms spread and matured [12].

Pakistan: The locust outbreak had been affecting eastern Pakistan since June 2019 [13]. In November 2019, Karachi experienced its first locust attack since 1961.

On January 29, 2020, the provincial Khyber Pakhtunkhwa government declared an emergency in nine southern districts of the province to stop the locusts from spreading

[14]. The districts of Dera Ismail Khan, Tank, Lakki Marwat, Bannu, Karak, Kohat, Hangu, and North and South Waziristan were the ones to declare an emergency.

On February 1, 2020, the Pakistani government decided to declare a national emergency in order to assist farmers and protect crops [15].

India: The swarms in India came from Iran and Pakistan, but pesticides and specialized equipment have brought the situation under control. Albeit the degree of the harm is to be evaluated, there was no significant misfortune [16]. Various convenient measures and a shift in twist course had forestalled a spread and enormous scope harm to the rapeseed and cumin seed crops, the authorities said. The episode started late 2019 in Gujarat and Rajasthan.

1.1.4. Effect of Climate

Environmental change is a vital driver of the flow flare-up. Swarms of desert locusts have been seen in the area for centuries, but scientists say that unusual weather caused strong cyclones and heavy rains in the Arabian Peninsula. This caused more vegetation to grow than usual, making it ideal for locusts to eat and multiply. It is anticipated that rising sea temperatures will result in more extreme rainfall, which will make it possible for eggs to hatch and eggs to reproduce. The Swarm-dispersing cyclones are becoming stronger and more frequent [1].

1.2. Motivation

Pakistani farming terrains have been confronting exceptionally basic assaults of beetles. The result of these attacks is starvation. However, locust attacks have not been prevented since 1990.

During the last ≈100 years, Pakistan was tormented by significant desert grasshopper attacks in 1926, 1952, 1962, and 1992. Pakistan's preparedness decreased during the lull as a result of competing demands because the 1992 invasion occurred 27 years before the most recent event. According to FAO 2020d, Pakistan's Space and Upper Atmosphere Research Commission designated desert locust-prone areas in response to the most recent incident based on vegetation distribution, soil type, desert locust history, and climate conditions [17].

According to FAO 2020c, at least 161,720 km2, or 36.9%, of Pakistan is susceptible to desert locust infestation. Swarms that had formed in the region of Rub al Khali in Saudi Arabia began making their way west across the Red Sea to the Horn of Africa, south into Yemen, and east to Iran by the end of 2018 (Showler et al.). 2021). By Walk 2019, swarms spread to Pakistan, and, in East Africa, into Kenya and Uganda (FAO 2019, FAO 2020e, Tong 2021). Infesting crops in the Dasht, Kund Mesori, Mastung, and Quetta

districts by June, swarms first entered Pakistan's Baluchistan Province (Kalair 2020) and damaged cotton, Gossypium hirsutum L., in the Khairpur, Ghotki, and Sukkur districts of Sindh Province (The Nation News 2019) [18].

Pesticide use is bad for crops and the environment. In order to keep locusts away from farms, we offer a solution in the form of an electric field. The agricultural land is safeguarded by this solution.

The federal government's Plant Protection Department, which was in charge of stopping the spread of locusts, was facing a major setback when 18 aircraft that had been used to spray aerially to stop locust attacks stopped working.

1.3. Problem Statement

We are aware that Pakistan is an agricultural nation. Our economy is primarily based on agriculture, and locusts are a terrible threat to our herbaceous vegetation. Locusts are migratory grasshoppers that can grow quickly and travel in large groups. In addition, when the population density is high, the individuals exhibit distinct behavioral states, color, and body shapes. to safeguard our nation from starvation, our farms from locusts, and the environment from pollution.

An eco-friendly operating system is being developed by us.

The procedure will:

- Identify the locusts' presence.
- Start the electric field.
- Get rid of the locusts.

1.4. Electric Fence

We, first and foremost, need to comprehend what an electrified barrier is. A wall can be characterized as a construction that fills in as a boundary and is normally associated with wires. An electrified barrier is a hindrance that utilizes electric shocks to hinder individuals or potentially different creatures from crossing a limit [19]. The voltage of the shock might have impacts going from inconvenience to death. Most electric walls are utilized for rural fencing and different types of non-human creature control.

A two-wire system can be used in places with poor soil and poor earth grounding conditions, with one wire acting as an electrical ground and the other as an earth ground. When alternating "hot" and ground wires are installed, this two-wire electric fence system is used for different purposes.

It is forbidden to allow the electrically charged fence wire to continuously come into contact with shrubs, tall grass, fence posts, nails, or any other conductive objects. In any case, the electric charge from the wall wire will lose its "stunning" power.

1.4.1. Design and function

When an animal touches an electric fence, an electrical circuit is completed. A power energizer is a component that transforms power into a brief high-voltage pulse. A roughly one-second electrical pulse is sent along a connected bare wire from one terminal of the power energizer. A ground or earth rod is a metal rod that is embedded in the ground and connects to another terminal. During a pulse, an animal that touches the earth and the wire will complete an electrical circuit and conduct the pulse, resulting in an electric shock. The voltage, pulse energy, degree of contact between the recipient and the fence or ground, and path that the current takes through the body all affect the shock's effects; It can be unpleasant, painful, or even fatal, depending on the severity**.**

An electrified barrier is more affordable than a wooden one, however, it is less solid and needs more upkeep. Notwithstanding those inconveniences, it tends to be exceptionally valuable when joined with a wooden wall. It keeps ponies from gnawing the salt-impregnated walls and from pushing the rails. Additionally, the animals will continue to be protected in the event of an electric outage.

1.4.2. Fencing materials

The most common material for electric fences is smooth steel wire, which can be thin and fine for a single line or thicker for a high-tensile (HT) fence. Electrifying woven wire or barbed wire fences is a less common option, but it creates a more dangerous fence, especially in the event that an animal gets caught in the material. The electrified fence itself must be shielded from the elements and from any materials that could conduct electricity and cause the fence to catch fire or short out. Therefore, fencing cannot be attached directly to wood or metal posts and must avoid vegetation. Plastic or porcelain insulators are typically attached to wooden or metal posts that are inserted into the ground, or plastic posts are utilized. The posts are then attached to the conducting material.

1.5. Applications

1.5.1. Use in Agriculture

Many agricultural areas use permanent electric fencing because it can be constructed at a lower cost and in a shorter amount of time than traditional fences (it uses plain wire and is much lighter because it does not need to physically restrain animals). When compared to fences made of barbed wire or certain kinds of woven wire that have large openings

that can entangle the feet, there is a lower risk of injury to livestock, particularly horses. In practice, most animals tend to avoid the fence for a long time even when it is inactive once they learn the unpleasant consequences of touching it. However, some animals learn to avoid the shock by pushing other animals through the fence or quickly running under the fence between pulses.

Temporary fencing that supports managed intensive grazing (also known as rotational or "strip" grazing) is constructed using electric materials as well. In some places, it is also used to confine horses and pack animals for the night while trail riding, hunting, or competing in endurance and competitive trail riding events. Commonly, at least one strand of wire, engineered tape or line is mounted on metal or plastic posts with stakes at the base, intended to be crashed into the ground with the foot. For a hand-fixed brief wall of zapped rope or web in a little region, these are typically divided at something like 12 to 15 feet (around four meters) to keep the fencing material from drooping and contacting the ground. Step-in posts may be able to be placed at greater distances without the risk of the fencing material sagging in larger areas where tools are used to stretch the wire.

We will utilize this electrified barrier against insect assault which is an extreme issue for our farming.

1.5.2. Buried electric fences

Dogs and livestock can sometimes be contained by using buried electric fences, which are also known as "invisible fences" or "electronic fences." The animal's collar picks up a weak radio signal that comes from the buried wire. In the vicinity of the wire, the collar makes a warning sound that, if ignored, causes a slight shock. People and different creatures know nothing about the covered line. The collar uses GPS signals to determine proximity to a predetermined "virtual fence" in a similar system that does not require a physical installation [20].

CHAPTER NO. 2

LITERATURE REVIEW

In the last few decades, several methods have been adopted throughout the world to stop Locust's invasion in the farm fields. Some of them are explained below:

2.1. Previous Work

Pesticides remained the most effective method of control. They were harmful to human health and the environment. As a consequence of this, there had been an increase in the amount of interest in biological control strategies that involved introducing natural pathogens or predators to the locusts that were being targeted without causing harm to other species.

Swarms of locusts move quickly and with unpredictability. Because of this, controlling their expansion and spread becomes difficult. Pesticides, including bio pesticides, must be used safely and effectively through aerial and ground spraying to stop their spread and reduce their propagation. To develop integrated pest management plans and assist nations in carrying out safe and efficient control operations, the World Bank is working closely with the FAO [21], which is the world's leading technical agency for desert locust control, as well as other partner organizations and nations.

Historically, people were unable to protect their crops from locusts, though they may have compensated by eating the insects. By the beginning of the 20th century, efforts were being made to disrupt the development of the insects by cultivating the soil in which they laid their eggs, collecting hoppers with catching machines, killing them with flamethrowers, trapping them in ditches, and crushing them with rollers and other mechanical tools. The organochloride dieldrin was discovered to be an extremely effective insecticide in the 1950s; however, due to its persistence in the environment and accumulation in the food chain, it was eventually outlawed in the majority of nations.

When locust control is required, tractor-based sprayers apply water-based contact pesticides to the hoppers early. This is efficient, but it takes a lot of time and effort; Concentrated insecticide that is sprayed over the insects or vegetation from an aircraft is the preferred method. To quickly treat large areas of land, ultralow-volume spraying of contact pesticides from aircraft in overlapping swaths is effective against nomadic bands.

Other ancient methods of avoiding locust attacks include the following:

- **Fire**: To erect a barrier or eliminate the locusts' primary food source, ancient societies would set fire to extensive areas of vegetation. The flames and smoke would stop the insects from moving forward or killing them.

- **Noise**: By beating drums, banging cymbals, or playing other instruments, people would make a cacophony of loud noises. The loud noises would cause the locusts to become disoriented and flee.

• **Catching**: To capture and eliminate locusts, ancient farmers would set traps. The locusts would fall into these pits or trenches, which had smooth, vertical walls, and struggled to escape. After that, they might be killed or driven out of the area.

• **Plant rotation**: Changing the establishing designs and turning crops occasionally could disturb the beetles' life cycle. Farmers may be able to prevent locusts from finding suitable breeding grounds by planting different crops at different times because locusts typically lay their eggs on bare ground.

• **Natural control**: Certain natural enemies of locusts, like birds or predatory insects, would eat the insects, according to some ancient civilizations. By either creating habitats for these natural predators or attracting them to the area, farmers would encourage their presence.

• **Removal by hand**: Locusts would be manually collected and removed from the affected areas by people. This could be accomplished by handpicking them or by removing them with nets or brushes.

• **Applying repellents**: To keep locusts away, a variety of substances were used as repellents. Smoke from burning particular plants, strong-smelling spices or herbs, and even substances like fish oil or animal dung are examples.

It is essential to keep in mind that even though these ancient methods were utilized with some degree of success—they were effective to some extent—they also result in noise, air pollution, and other problems. Pesticides also have negative effects on human health [22].

Locust control methods and technologies have developed significantly in recent years. To combat locust swarms more effectively today, monitoring systems, early warning networks, and aerial insecticide spraying are utilized.

2.2. Efforts by Pakistan

Desert locusts eat a wide range of crops and can be found in Africa, the Middle East, and southwest Asia. Swarms of locusts ride the winds and travel more than 100 kilometers per day during an outbreak, causing damage to approximately 60 nations and 20% of the world's land area.

Pakistan is traversed by locusts from both the west and the east. This region experienced a significant assault in 1926, before Pakistan's establishment in 1947. Another terrible outbreak of similar proportions took place in 1952. In 1962, desert locusts severely affected rice, wheat, and sugarcane crops. The horde destroyed more than 29 million square kilometers in 60 Afro-Asian nations in 1963, but it returned in 1967 and 1968.

The subsequent outbreak reached Pakistan in 1978 when the first wave of locust attacks in the desert and cultivated areas of south-western Pakistan in June affected an area of approximately 5500 km2. The subsequent attack occurred in 1992, followed by the one in 2019. Iran from the west and India from the east are the entry points for locusts into Pakistan [23].

2.2.1. During COVID-19

At the end of 2019, Pakistan experienced a particularly severe outbreak of desert locusts that affected approximately 18 million hectares, making it the largest and worst outbreak ever recorded (World Bank: Data for June 2020). Furthermore, this locust outbreak and COVID-19 have combined to severely damage Pakistan's agricultural sector, the country's primary industry, since 2020 [23].

Due to persistent drought and inflation, agricultural supplies like seeds and fertilizers have increased dramatically in Pakistan. The government declared a national emergency in February of last year in response to predictions of an attack by a swarm of desert locusts from East Africa and the Middle East.

2.2.2. Aid from JICA

Together with FAO, the Japan International Cooperation Agency (JICA) assists Pakistani farmers who have been harmed by locusts. Together with the Food and Agriculture Organization of the United Nations (FAO), JICA is developing a system to support agriculture by enhancing the capabilities of government officials who carry out pest control activities and the livelihoods of small-scale farmers affected by locusts. To strengthen Pakistan's ability to combat desert locusts, FAO has collaborated with the relevant government departments.

In response to this circumstance, JICA provided approximately 3,000 small-scale farmers in four provinces of Balochistan and three provinces of Punjab, where it collaborates in the field of agriculture, with emergency assistance in the form of wheat seeds and the necessary fertilizer. Wheat is a staple food for the people of Pakistan. The initiative to safeguard farmers' livelihoods from locust damage is currently underway [23].

2.2.3. Enhancing locust control and monitoring capabilities to prepare for future attacks

Beginning in March of this year, JICA, in collaboration with FAO and the Pakistani government, began enhancing the cooperative structure for small-scale farmers affected by locust. The project aims to improve small-scale farmers' nutrition and food security in the affected areas over the next year and to improve desert locusts' ability to be monitored

and controlled. This includes the creation of a pest control information network and GPS device training [23].

2.3. Efforts by Other Countries

The outbreak is rapidly developing. Nations have been affected as of the middle of April 2020: 11 in North Africa and the Middle East, 3 in South Asia, and 9 overall in East Africa. According to a recent joint impact assessment report produced by the Ethiopian government and the Food and Agriculture Organization (FAO), the desert locust outbreak in Ethiopia alone has resulted in the loss of 356,286 metric tons of cereal, the destruction of 197,163 ha of cropland, and the loss of 1,350,000 ha of pasturelands, leaving over a million Ethiopians in need of food assistance [1].

2.3.1. Help from World Bank

A program worth 500 million dollars has been approved by the World Bank Group to provide adaptable assistance to African and Middle Eastern nations affected by the locust outbreak. Through cash transfers and other social protection measures, the priority is assisting affected households in meeting their immediate food requirements and safeguarding their physical and human capital assets. In order to get communities back on their feet, follow-up actions will focus on restoring food production and livelihood systems, as well as strengthening national surveillance and early warning systems to reduce the risk of future outbreaks.

Djibouti, Ethiopia, Kenya, and Uganda are the "first-mover" nations that will receive funding in the initial phase of the program. Together, these nations will receive a total of US$160 million in funding, making them one of the hardest hit countries. Locust response efforts in Pakistan (US$ 200 million), Somalia (US$ 40 million), and Yemen (US$ 25 million) also receive funding[1].

CHAPTER NO. 3

ELECTRIC FENCE FOR LOCUSTS

3.1. Proposed Solution

Our solution has undergone significant improvements to enhance its efficiency in addressing the locust infestation problem. We have implemented an effective method of delivering an electric shock to locusts, ensuring better results. Additionally, we have incorporated a sensitive voice module V3 into our device, enabling it to detect the presence of locusts based on their sound. This module is trained by inputting specific sound patterns into it, which are then recognized as commands. Once the module senses the presence of locusts, an alarm is triggered, signaling the need for action.

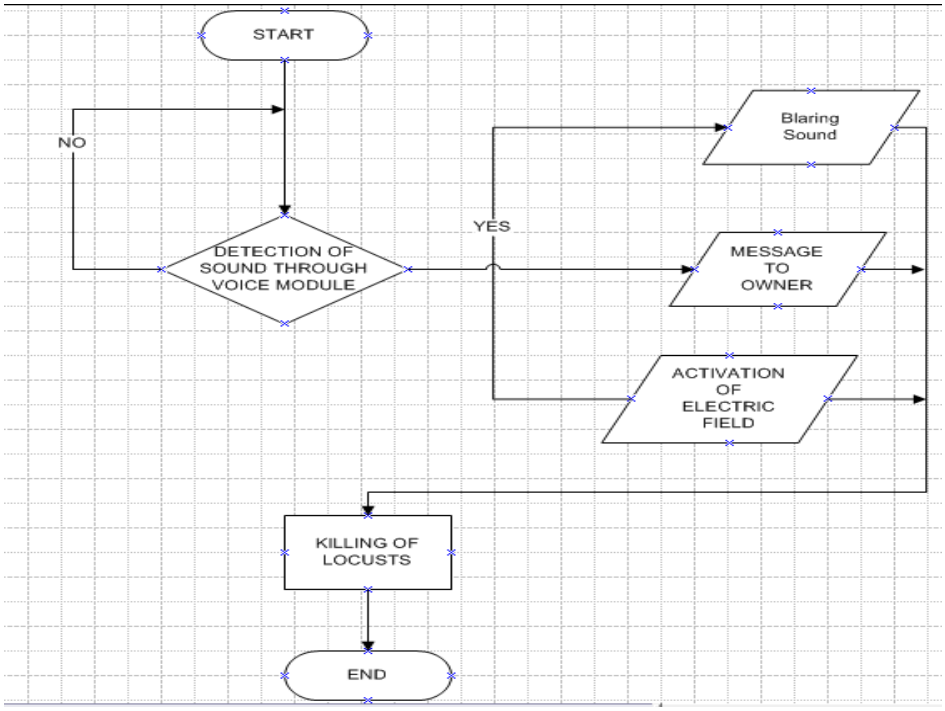

Fig 1: Flowchart

3.1.1. Working of Voice Module

The key enhancement lies in our device's ability to detect locusts based on their sound. By incorporating a sensitive voice module V3, we have provided the device with the capability to recognize and differentiate locust sounds from other ambient noises. The training process involves connecting the module to an Arduino and feeding it with a variety of locust sound samples. This enables the module to learn and identify the distinct sound patterns associated with locusts.

Once the module is trained, it becomes an integral part of our solution. When deployed in locust-prone areas, the device continuously monitors the surroundings for locust sounds.

As soon as the module detects the characteristic sound patterns, it triggers an alarm. This alert notifies the operators or farmers about the presence of locusts in the vicinity, allowing them to take immediate action.

To effectively deter and control the locust population, our solution incorporates an electric fence. The ZVS flyback driver plays a crucial role in generating the necessary current to electrify the fence. Its zero voltage switching mechanism ensures efficient operation by minimizing heat generation and maximizing energy transfer.

3.1.2. Activation of Electric Field

To activate the electric fence, our device utilizes a ZVS (Zero Voltage Switching) flyback driver. This driver efficiently generates current in the field. As the locusts come into contact with the electric field lines, they experience an electric shock. This method proves to be more effective compared to previous solutions proposed for this issue. We have taken into consideration the understanding and perspective of the general public, ensuring that our solution is practical and easily comprehensible.

3.1.3. Working of the Whole Prototype

As the locusts approach the electric field lines, they encounter the electric current flowing through them. This current, while not harmful to humans or larger animals, delivers an electric shock to the locusts. This shock serves as a deterrent, effectively discouraging the locusts from crossing the fence and entering farmland or other sensitive areas. Compared to previously proposed solutions, our approach addresses the locust infestation problem with a practical and understandable framework. By incorporating a voice module for sound-based detection, we have increased the accuracy and reliability of locust identification. The integration of a ZVS flyback driver ensures efficient and effective utilization of electrical current in deterring locusts. Furthermore, our solution emphasizes simplicity and ease of use, making it accessible and feasible for farmers and operators with varying levels of technical expertise.

In summary, our enhanced solution utilizes an electric shock mechanism, voice module V3, and ZVS flyback driver to effectively detect and deter locusts. By considering the practical aspects and providing a comprehensive understanding, we aim to contribute to the successful management of the locust infestation problem.

3.1.4. Expected Result

This proposed solution results in stopping the locusts' invasion as they destroy the agricultural lands. When exposed to the electric field, they are killed by getting an electric shock and thus the farms are protected from the destruction.

3.2. Hardware Implementation

We have developed an electric fence using wooden strips to create a boundary and wires to form the actual fence. The purpose of this fence is to prevent locusts from entering the protected area. To power the fence, we have incorporated a ZVS flyback transformer, which supplies sufficient current to effectively eliminate the locusts. Additionally, we have integrated a voice module to detect the distinctive sounds produced by locusts. When the voice module detects these sounds, it triggers the activation of the electric field.

However, during the implementation process, we encountered several challenges, particularly in operating the voice module and connecting the ZVS driver to the fence to establish the electric field. Connecting the ZVS driver to the fence posed a challenge as it required careful consideration of the electrical connections and ensuring the safety of the operators. We needed to ensure that the connection points were properly insulated and protected to prevent any accidental contact with the electric field. Furthermore, appropriate measures were taken to prevent damage to the ZVS driver and to ensure the efficiency and effectiveness of the electric field. Operating the voice module presented its own set of challenges. It required proper calibration and sensitivity adjustment to accurately detect the unique sounds produced by locusts while avoiding false positives triggered by other ambient noises. Additionally, we had to optimize the placement and positioning of the voice module to maximize its effectiveness in detecting locust sounds throughout the protected area.

Throughout the development process, we worked on troubleshooting and fine-tuning the voice module's performance to achieve reliable locust detection. This involved adjusting the sensitivity settings, filtering out background noise, and implementing signal processing techniques to enhance the accuracy of locust sound detection. In summary, our project involved constructing an electric fence using wooden strips and wires, powered by a ZVS flyback transformer, to deter locusts. The integration of a voice module added an automated locust detection capability. However, challenges were encountered in connecting the ZVS driver to the fence and ensuring optimal performance of the voice module. Through careful calibration and troubleshooting, we aimed to achieve an efficient and effective locust prevention system.

3.2.1. Mathematical Values

Output Voltages= 2000V, Output Current= 15mA

Input Voltages= 215- 255 V, AC Input Current= 0.2Amperes

Average Power= 30 Watts , Frequency= 25- 100kHz [$f = 1 / (2\pi \sqrt{L C})$]

Electric Field (v/cm)	Observation
1000	From 1cm to 2cm it effects
1200	It burns the wings and legs of small locust
1500	It gives a good result on as small as 3cm Gregarious locust.
2000	It completely immobilized the locust.

3.3. Hardware Components

3.3.1. ZVS driver

A "ZVS driver" is an acronym for Zero Voltage Switching driver, which functions as a self-resonant oscillator. It is a relatively uncomplicated circuit that can generate powerful oscillations with minimal energy loss. Its primary application lies in producing the necessary sine wave required to drive high-frequency transformers, commonly found in LCD cold-cathode lamp drivers, among other uses.

The ZVS driver operates based on the principle of Zero Voltage Switching, which refers to the technique of switching a power device on or off when the voltage across it is zero. By employing this method, the driver minimizes energy dissipation during the switching process, enhancing its overall efficiency. This feature makes it an ideal choice for applications where power conversion with minimal losses is desirable. The ZVS driver circuit consists of several key components. It typically comprises a resonant tank circuit, which includes an inductor and a capacitor, connected in parallel. This tank circuit forms the backbone of the self-resonant oscillator. Additionally, the circuit includes power switches, often MOSFETs (Metal-Oxide-Semiconductor Field-Effect Transistors), that control the flow of power to the transformer. These switches are driven by a control mechanism, which ensures the switching occurs at zero voltage points.

The operation of the ZVS driver can be understood as follows: Initially, the power switches are turned on, allowing current to flow through the resonant tank circuit. The energy stored in the inductor and capacitor starts to increase. As the energy builds up, the voltage across the switches begins to rise. When the voltage reaches a certain threshold, the control mechanism triggers the switches to turn off. At this moment, the voltage across the switches is ideally zero, achieving Zero Voltage Switching.

With the switches turned off, the energy stored in the resonant tank circuit continues to circulate. The inductor and capacitor exchange energy, resulting in oscillations. This oscillating energy is then coupled to the transformer, where it is further transformed and utilized for specific applications, such as driving the CCFLs in LCDs. The ZVS driver's ability to oscillate a large amount of power with minimal loss makes it highly efficient in various high-frequency applications. It enables power conversion with reduced heat dissipation and improved overall performance. Furthermore, its simplicity in design and

operation makes it a cost-effective solution for generating the sine wave needed for high-frequency transformers.

3.3.2. ZVS (Zero Voltage Switching) flyback transformer driver

The ZVS (Zero Voltage Switching) flyback transformer driver, invented by Vladmiro Mazilli, is recognized as a highly efficient and powerful driver. It employs resonant switching to drive the flyback transformer, wherein the MOSFETs (Metal-Oxide-Semiconductor Field-Effect Transistors) are specifically designed to switch on or off when the voltage across them reaches zero.

By switching the MOSFETs when there is no voltage present, the ZVS driver generates minimal heat, with the only heat produced originating from the internal resistance of the MOSFETs. This is in contrast to simpler flyback drivers that use a 555 timer, as the ZVS driver allows for longer operation periods of the flyback transformer before the MOSFETs overheat. In fact, with high-quality MOSFETs, it may even be possible to run the ZVS flyback driver indefinitely, or until the circuit is interrupted.

In summary, the ZVS flyback transformer driver, developed by Vladmiro Mazilli, is known for its efficiency and power. Through its use of resonant zero voltage switching and careful MOSFET design, it minimizes heat generation, allowing for extended operation of flyback transformers without the risk of MOSFET overheating.

3.3.3. Working

A Zero Voltage Switching (ZVS) flyback transformer is a type of power converter circuit used in switch-mode power supplies (SMPS). It is designed to minimize switching losses and improve the overall efficiency of the power supply.

The basic working principle of a ZVS flyback transformer is as follows:

1. Input Stage: The input voltage is typically rectified and filtered to provide a DC voltage source. This DC voltage is then fed into a switching transistor (typically a MOSFET) controlled by a driver circuit.

2. Switching Stage: The driver circuit controls the switching transistor to alternately turn it ON and OFF at a high frequency (typically tens or hundreds of kilohertz). When the transistor is ON, current flows through the primary winding of the flyback transformer, storing energy in its magnetic field.

3. Energy Storage: While the transistor is ON, the energy stored in the transformer's primary winding builds up in the form of a magnetic field. This energy is transferred to the secondary winding of the transformer when the transistor turns OFF.

4. Energy Transfer: When the transistor is turned OFF, the sudden removal of current causes the magnetic field in the primary winding to collapse rapidly. This rapid change in the magnetic field induces a voltage in the secondary winding of the transformer.

5. Output Stage: The induced voltage in the secondary winding is rectified and filtered to produce the desired output voltage. This voltage is then used to power the load.

The ZVS technique in a flyback transformer is employed to reduce switching losses during the ON and OFF transitions of the transistor. By ensuring that the switching occurs when the voltage across the transistor is close to zero, ZVS minimizes the power dissipated during the transition, leading to improved efficiency. ZVS is typically achieved by incorporating additional circuitry such as resonant capacitors and inductors, which help to shape the voltage and current waveforms to enable zero voltage switching. This soft switching technique reduces the stress on the components and improves efficiency by reducing switching losses. Overall, a ZVS flyback transformer offers benefits such as improved efficiency, reduced switching losses, and increased reliability in switch-mode power supply applications.

3.4. Voice Recognition Module

The Voice Recognition Module is a device that can recognize and process voice commands. It is designed to receive configuration commands and provide responses through a serial port interface. This module offers a convenient way to control various electrical devices, including cars, using voice commands.

3.4.1. Features

One of the key features of the Voice Recognition Module is its ability to store up to 15 voice instructions. These instructions are divided into three groups, with five instructions in each group. Before using the module, it is necessary to train it by providing voice instructions for each group. It's important to note that this module is speaker dependent, meaning that it is specifically trained to recognize the voice of the individual who trains it. As a result, if someone else tries to use the module without training, it may not work effectively for them.

The Voice Recognition Module boasts an impressive recognition accuracy of 99% under ideal conditions. This accuracy rate ensures that the module can reliably understand and respond to voice commands, providing a smooth user experience.

3.4.2. Operating Voltages

In terms of technical specifications, the module operates within a voltage range of 4.5V to 5.5V and consumes less than 40mA of current during operation. It offers both a digital interface, which includes a 5V TTL level UART interface and GPIO (General Purpose Input/Output) pins, as well as an analog interface consisting of a 3.5mm mono channel microphone connector and a microphone pin interface.

The physical dimensions of the Voice Recognition Module are compact, with a size of 30mm x 47.5mm. This compact form factor allows for easy integration into various devices and systems.

Overall, the Voice Recognition Module provides a reliable and efficient solution for voice-controlled operations. Its ability to store multiple voice instructions, coupled with its high recognition accuracy, makes it an ideal choice for controlling electrical devices, such as cars, using voice commands.

3.5. Relay

Relay modules are circuit boards that house one or more relays, typically rectangular in shape with 2, 4, 8, or even up to 16 relays mounted on them. These modules also incorporate other components such as indicator LEDs, protection diodes, transistors, resistors, and more.

A relay, which is the main component of the module, is an electrical switch used to control devices and systems that operate at higher voltages. In the case of a relay module, the switching mechanism is typically based on an electromagnet [24].

The input voltage for relay modules is usually DC, while the electrical load that a relay controls can be either AC or DC, within the limits specified by the relay's design. Various options of input voltage ratings are available for relay modules, such as 3.2V or 5V for low power switching, or 12V or 24V for heavy-duty applications

Important information about the relay module, including input voltage rating, switch voltage, and current limit, is typically printed on the surface of the device for easy reference [25]. This allows users to identify and select the appropriate relay module for their specific application.

3.5.1. Working

The working principle of a relay module is straightforward. It utilizes an electromagnet to control the opening and closing of electrical contacts. Here is a step-by-step explanation of how relay modules function:

- A typical relay module consists of an input side with 3 or 4 jumper pins and an output side with 3 screw terminals.
- When a control signal is applied to the input side, it activates the electromagnet, causing an armature to be attracted.
- This action closes the switch contacts on the output side, allowing electricity to flow and power the connected device or system.
- To protect the relay module circuit and the input device from flyback voltage, a diode known as a flyback diode is often connected in parallel with the electromagnet coil. The flyback diode permits current flow in only one direction.
- For enhanced isolation, an optocoupler may be used. In an opto-isolated relay module, a photoelectric device on the input side controls the switching action of the electromagnet.
- Relay modules are available in two switch configurations: normally open (NO) and normally closed (NC).
- In a NO switch, the contacts are open when the electromagnet is inactive and close when it is activated.
- In contrast, an NC relay switch remains closed by default and opens only when the relay is activated.

A relay is an electromagnetic switch that is widely used in various electrical and electronic systems to control the flow of electric current. It operates based on the principle of an electromagnet.

Here's a simplified explanation of how a relay works:

1. Electromagnet: A relay consists of a coil, usually made of a copper wire, wound around a core. When an electric current flows through the coil, it generates a magnetic field around the core. This coil and core assembly is called an electromagnet.
2. Contact Points: The relay has one or more sets of switch contacts that are mechanically linked to the electromagnet. These contacts can be normally open (NO) or normally closed (NC).
    a. Normally Open (NO): In this configuration, the contacts are open when the relay is not energized, and they close when the relay is energized.
    b. Normally Closed (NC): In this configuration, the contacts are closed when the relay is not energized, and they open when the relay is energized.
3. Control Circuit: The control circuit provides the current necessary to energize the relay's coil. When a voltage is applied to the coil, it creates a magnetic field, causing the contacts to move from their default position.
4. Switching Action: When the coil is energized, the magnetic field attracts a metal armature or an iron core. This action moves the contact points, causing them to

change their state. For example, if the relay has normally open (NO) contacts, they will close when the coil is energized, completing the circuit and allowing current to flow through the contacts. If the relay has normally closed (NC) contacts, they will open when the coil is energized, breaking the circuit and stopping the current flow [26].

5. Application: Relays are used in various applications, such as controlling electrical motors, switching high-power circuits, interfacing low-power electronic devices with high-power loads, and providing galvanic isolation between different circuits [27].

3.5.2. Advantages of using Relay

Relays offer advantages such as electrical isolation between the control circuit and the switched circuit, the ability to switch high currents or voltages with a low-power control signal, and the capability to provide multiple sets of contacts for different switching configurations.

It's important to note that the actual construction and design of relays can vary depending on the specific application and requirements. Advanced relays may incorporate additional features like diodes for back-EMF protection [28], latching mechanisms for maintaining their state without continuous power, or solid-state components for faster switching.

3.6. Arduino

The well-known open-source hardware and software company Arduino focuses on developing microcontrollers and microcontroller kits for digital device construction. The project operates as a collaborative community and offers its hardware under a CC BY-SA license, allowing anyone to manufacture Arduino boards, while the software is licensed under the GNU Lesser General Public License (LGPL) or the GNU General Public License (GPL), enabling software distribution [29].

Arduino boards employ various microprocessors and controllers and provide a range of digital and analog input/output (I/O) pins. These pins can be connected to expansion boards known as "shields" or breadboards for prototyping and interfacing with other circuits. The boards are equipped with serial communication interfaces, including options like Universal Serial Bus (USB) on specific models, which facilitate program loading. Programming for Arduino microcontrollers is typically done using the Arduino. Programming Language, an API based on C and C++, and a modified version of the Processing IDE [30]. The Arduino project offers an integrated development environment (IDE) and a command-line tool, making it accessible to both novice and professional users.

The inception of Arduino traces back to 2005 when it was originally developed for students at the Interaction Design Institute Ivrea in Italy. The project aimed to provide an affordable and user-friendly solution for individuals, regardless of their skill level, to create interactive devices utilizing sensors and actuators. This initiative catered to hobbyists, allowing them to construct simple robots, thermostats, motion detectors, and more.

The name "Arduino" was inspired by a bar in Ivrea, Italy, frequented by the project's founders. The bar shared its name with Arduin of Ivrea, who served as the margrave of the March of Ivrea and was also the King of Italy from 1002 to 1014. This historical connection lent itself to the name chosen for the project [31].

Overall, Arduino has become synonymous with accessible and versatile microcontroller platforms, empowering users to bring their creative ideas to life through digital devices and automation. Its open-source nature, extensive community support, and range of available resources have made Arduino a popular choice for both educational and professional projects.

3.6.1. Working

Arduino is an open-source electronics platform based on easy-to-use hardware and software. It consists of a microcontroller board and a development environment that allows users to write, program, and upload code to control various electronic components and devices. Here's a simplified explanation of how Arduino works:

1. Hardware: The Arduino board is the physical component of the platform. It typically includes a microcontroller, digital and analog input/output pins, power supply connectors, and other components. The microcontroller is the brain of the Arduino and is responsible for executing the uploaded code.

2. Development Environment: The Arduino development environment is a software tool that runs on a computer. It provides an integrated development environment (IDE) for writing, editing, and compiling code. The IDE also allows users to upload the compiled code to the Arduino board for execution.

3. Programming Language: Arduino uses a simplified version of the C/C++ programming language. Users can write code in the Arduino IDE, which provides a set of libraries and functions that abstract low-level operations, making it easier to control and interact with the Arduino's hardware components.

4. Code Execution: Once the code is written, it can be compiled and uploaded to the Arduino board via a USB connection. The code is stored in the microcontroller's memory. Upon power-up or reset, the microcontroller starts

executing the uploaded code, following the instructions and controlling the connected components accordingly.

5. Input/Output: The Arduino board has a variety of input and output pins that can be used to connect and control external devices or sensors. These pins can be configured as digital inputs or outputs or as analog inputs. Digital pins can read or output binary states (high or low), while analog pins can read analog voltages.

6. Libraries and Functions: Arduino provides a rich set of libraries and functions that simplify common tasks, such as reading sensor values, controlling motors, communicating with other devices (e.g., using protocols like I2C, SPI, or serial communication), and interacting with displays or user interfaces.

7. Customization and Expansion: Arduino's modular and open-source nature allows users to customize and expand its functionality. Additional shields or modules can be connected to the Arduino board to add specific features or capabilities, such as wireless communication, GPS, or motor control.By leveraging the Arduino platform, users can create a wide range of projects, from simple LED blinking experiments to complex robotics systems or home automation setups. The simplicity of the hardware and software, along with the vast community support and documentation, makes Arduino accessible to beginners and experienced users alike [32].

3.7. Software implementation

To detect the presence of locusts, we are using an Arduino UNO, a mike and a voice recognition module V3. Connect the UART pins RX and TX to the digital ping of the Arduino D4 and D5.

Selecting the Hardware

Choose a suitable hardware platform for your voice module. Options may include microcontrollers like Arduino or Raspberry Pi, which offer analog or digital input capabilities.

Audio Sensor

Select an audio sensor that can capture and convert sound waves into electrical signals. A popular choice is a small electret microphone module, which is affordable and readily available.

Circuit Connection

Connect the audio sensor to the chosen microcontroller. Usually, the audio sensor outputs an analog signal, so you will need to use an analog-to-digital converter (ADC) if your

microcontroller doesn't have built-in ADC functionality. Follow the datasheets and documentation of your selected components for proper wiring.

Programming Environment

Set up the programming environment for your chosen microcontroller. Install the necessary software development tools and libraries to facilitate programming and communication with the audio sensor.

Sound Detection Algorithm

Design a sound detection algorithm to identify locust sounds. This algorithm should analyze the audio input from the sensor and determine if it matches the characteristic frequency patterns of locust sounds. You can use signal processing techniques such as Fast Fourier Transform (FFT) to extract frequency components.

Threshold and Validation

Set a threshold for sound intensity to distinguish locust sounds from background noise. Experiment with different threshold values to ensure accurate detection. You may also implement additional validation criteria, such as the duration of the sound or specific frequency patterns, to reduce false positives.

Integration and Feedback

Integrate the sound detection algorithm into your microcontroller code. Upon detecting a locust sound, you can trigger various actions like activating an alarm, sending a notification, or interfacing with other systems for locust control purposes.

Testing and Refinement

Test the voice module in different environments to ensure its effectiveness in detecting locust sounds. Make adjustments to the algorithm or threshold values if necessary, based on the test results.

Deployment and Continuous Improvement

Deploy the voice module in the desired locust monitoring areas. Monitor its performance and gather feedback to identify any potential improvements or optimizations for future iterations.

Code

The Code we used is in C language run on Arduino IDE and is given in Appendix.

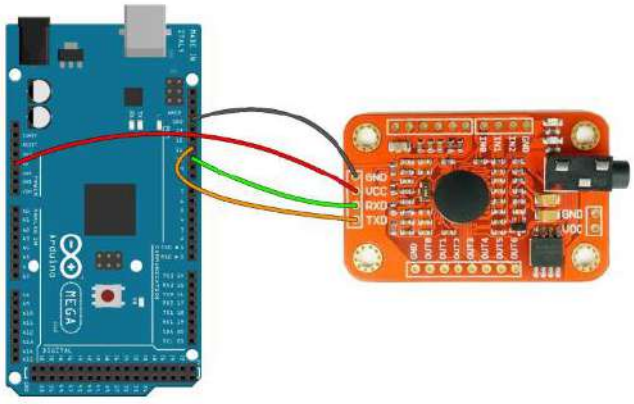

Fig 2: Arduino and Voice module V3

3.8. Training of voice commands

Before connecting the module with the Arduino, these are the steps we have to follow:

- Firstly, download and install the "voicerecognitionv.3.h" Arduino library.
- Connect the circuit to the computer and launch the Arduino IDE.
- Select the right Arduino board. (Tools -> Board) and then select the correct COM port. (Tools -> Port)
- Open the program for training the module [33].

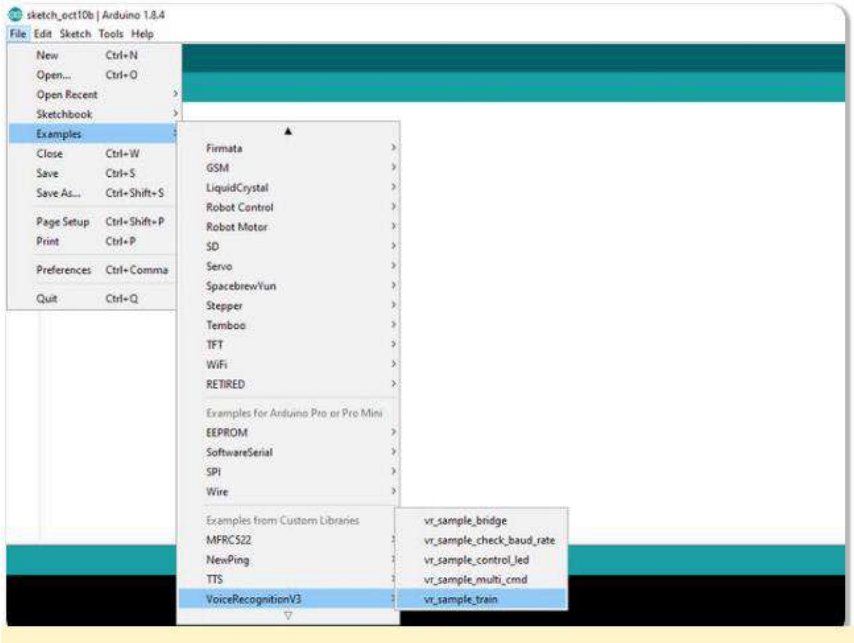

Fig 3: Arduino software

3.8.1. Add voice command

1. To use a voice module, you need to first upload the code to your Arduino. After that, you can open the Serial Monitor and select the baud rate to 115200. A menu will appear on the Serial Monitor, and you can use the "train" command to train the module. The V3 can store up to 80 voice instructions, each of which can last for 1500 milliseconds. Every command is stored in a specific location with an address ranging from 0 to 254.
2. When using the "train" command, you need to include the address in the command to store the voice command in a specific location. The command has the following syntax: "train address Train 0; Train 20; Train 254". To operate the LED, you need to enter two spoken instructions to turn it on and off.
3. To train the module, you can enter the command and the address you want to save it at in the serial monitor. For example, you can enter "train 20" to store a command at address 20. After entering the command, wait for the "speak now" message to display on the serial monitor. Speak clearly and loudly enough into the microphone to enter your instruction for the LED to turn on.

4. If the instruction is understood, a second message will appear requesting you to repeat it. You can repeat the phrase to make the instruction heard. If you were able to successfully load both instructions, you can now upload the code for controlling the LED.
5. Overall, the process of using a voice module involves uploading the code to Arduino, opening the Serial Monitor, training the module using the "train" command, storing voice commands in specific locations, and entering spoken instructions to operate the LED. With these steps, you can successfully use a voice module to control an LED.

CHAPTER NO 4

RESULT

The project's primary objective is to evaluate the efficacy of electric fencing as a method to combat locust infestations and minimize crop damage. The bugs or locusts will get a shock when they try to cross the boundary of the electric field around the farms. These dead pests can be used as fertilizers for the crops. In this way, our farms are protected through locusts and these locusts are further used as natural fertilizers which are very beneficial for efficient production and quick growth.

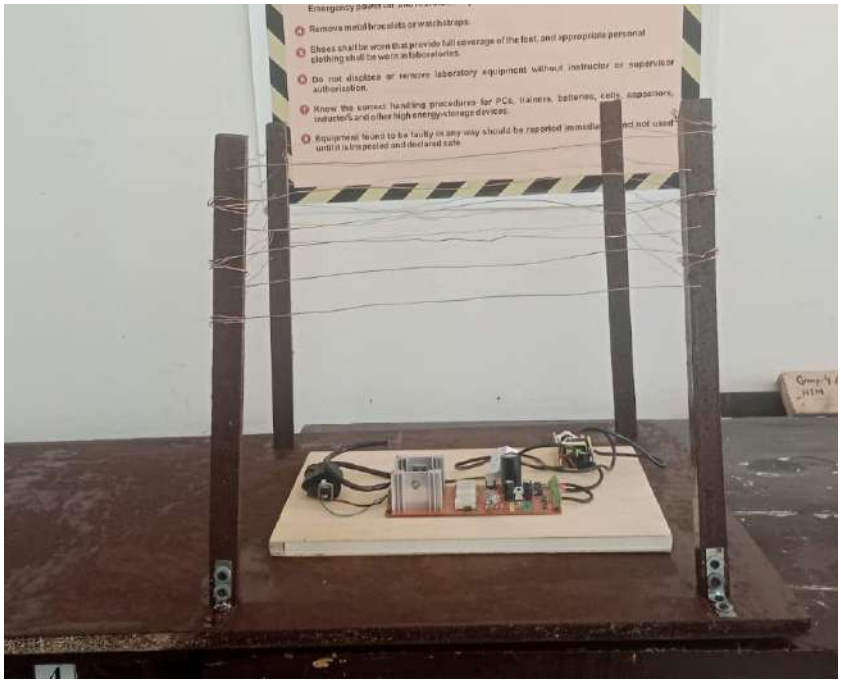

Fig 4: Electric Fence Prototype

The research yielded the following significant outcomes:

4.1. Reduction in Locust Incursions

The implementation of electric fencing proved highly effective in deterring locusts from entering agricultural fields. By creating a physical and psychological barrier, electric fencing successfully prevented locusts from infiltrating protected areas.

4.2. Behavioral Responses

Locusts exhibited distinct avoidance behaviors when confronted with electric fences. Upon encountering the electrified barrier, locusts altered their flight paths and changed directions to avoid contact. This behavior showcased the potential of electric fencing in redirecting locust movement away from vulnerable crops.

4.3. Fence Design and Configuration

The study underscored the critical role of fence design and electrical configurations in maximizing the efficiency of electric fencing. Factors such as fence height, wire spacing, voltage levels, frequency, and pulse duration significantly influenced the deterrent effect

on locusts. Optimal designs emphasized creating a robust and consistent electrical barrier without causing harm to the locusts.

4.4. Crop Damage Reduction

The implementation of electric fencing resulted in a notable decrease in crop damage inflicted by locusts. By effectively preventing locusts from accessing agricultural fields, farmers experienced reduced losses and improved crop yields. Electric fencing emerged as a proactive measure to safeguard crops and enhance overall agricultural productivity.

CHAPTER NO. 5

DISCUSSION

5.1. Discussion

We want the development of an Eco-Friendly Electric Fence System for locust control in Pakistan's agricultural sector because we know that Pakistan heavily relies on agriculture as the backbone of its economy. However, the country faces a significant threat to its herbaceous vegetation from locusts, migratory grasshoppers that can rapidly grow in population and cause widespread damage to crops. The conventional use of pesticides poses risks to both crops and the environment. To address this challenge and protect the nation from potential starvation, a novel eco-friendly operating system is being developed, centered around an electric fence system.

5.1.1. Electric Fence System

An electric fence is a barrier that utilizes electric shocks to deter individuals or animals from crossing a boundary. The system consists of wires connected to a power energizer, which converts power into high-voltage pulses. When an animal comes into contact with the electrified fence and the ground, it completes an electrical circuit and receives an electric shock. The intensity of the shock depends on various factors such as voltage, pulse energy, and the path of the current through the body. The electric fence system operates by completing an electrical circuit when an animal touches the fence. The pulses from the power energizer generate an electric shock, deterring the animal from crossing the boundary. The fence is typically made of smooth steel wire, either thin for a single line or thicker for high-tensile fences. The electrified fence must be insulated from materials that could conduct electricity, and plastic or porcelain insulators are used to attach the fence to wooden or metal posts.

Our Electric Fence system consists of a flyback transformer which is used to activate the electric field. A voice module is used to detect the presence of a locust's swarm. When a locust comes into contact with the electrified fence, the electric circuit is completed. This triggers the ZVS circuit, which initiates the generation of high-voltage pulses by the flyback transformer. The locust receives an electric shock upon contact with the fence, deterring it from crossing the boundary.

5.1.2. Comparative Analysis

Compared to conventional control methods, electric fencing demonstrated several advantageous aspects. It proved to be cost-effective, environmentally friendly, and relatively easy to install and maintain. Furthermore, electric fencing showcased comparable or superior performance in mitigating locust infestations, presenting itself as a viable alternative or complementary solution to existing control measures.

The findings of this project underscore the potential of electric fencing as a practical and efficient strategy for managing locusts in agricultural settings. These research outcomes

offer valuable insights to farmers, policymakers, and agricultural stakeholders who seek sustainable and effective approaches to combat locust infestations and protect crop yields.

However, it is crucial to emphasize the need for further research to evaluate the long-term effectiveness of electric fencing, optimize its design and configuration for different locust species, and assess its performance under diverse environmental conditions. Additionally, collaborative efforts are essential to raise awareness, disseminate knowledge, and promote the adoption of electric fencing as a viable tool in locust control strategies.

5.2. Advantages and Impacts

The proposed electric fencing system offers several advantages and impacts:

5.2.1. Eco-friendly Solution

Unlike conventional pesticide spraying, the electric fencing system does not harm crops or the environment. It provides a sustainable and environmentally friendly approach to locust control.

5.2.2. Cost-effective and Efficient

Electric fences are cost-effective compared to aerial spraying of pesticides. Once installed, the system requires minimal maintenance and provides continuous protection against locusts

5.3. Design Issues

When implementing an electric fence to deter locusts, you may encounter several challenges or errors that can affect its effectiveness. Here are some common issues you might face:

5.3.1. Insufficient Voltage

If the electric fence does not generate a high enough voltage, it may not provide an effective deterrent for locusts. Low voltage levels can fail to repel the insects or allow them to breach the fence. Due to this reason we use ZVS circuit and Fly Back transformer for high voltage.

5.3.2. Inadequate Coverage

Locusts are capable of flying and hopping, so they can access crops or vegetation from various angles and heights. If the electric fence does not cover the entire area or if there are gaps, locusts can find entry points and bypass the fence's protection.

5.3.3. Grounding Issues

Proper grounding is crucial for the effectiveness of an electric fence. If the fence lacks adequate grounding, the electric current may not flow correctly, reducing its deterrent effect. Improper grounding can result in weak shocks or even no shock at all.

5.3.4. Vegetation Interference

Vegetation touching the electric fence wires can create a conductive path, diminishing the effectiveness of the electric shock. Overgrown plants, leaves, or branches should be regularly cleared away from the fence to maintain its efficiency.

5.3.5. Equipment Malfunction

Like any electrical system, components of the electric fence can experience malfunctions or failures over time. This may include issues with the power supply, wiring, transformer, or control circuitry. Regular maintenance and inspections are necessary to identify and address any equipment malfunctions.

5.3.6. Environmental Factors

Harsh weather conditions, such as heavy rain, strong winds, or extreme temperatures, can impact the performance of an electric fence. It may cause damage to the wires, insulators, or other components, leading to reduced functionality or complete failure. Remember to consider the limitations of the hardware and sound detection algorithm. Locust sounds can vary depending on species and environmental factors, so continuous monitoring and adjustment may be required to achieve accurate detection this is very critical error that we was facing during feeding the sound and testing the project performance.

5.3.7. Locust Adaptation

Locusts are resilient and adaptive creatures. They may learn to overcome the electric fence by developing strategies to avoid or tolerate the electric shocks. Continuous monitoring and occasional adjustments to the fence design or settings may be necessary to counter locusts' adaptive behaviors.

5.3.8. Human Error

Incorrect installation, improper maintenance, or inadequate training of personnel responsible for the electric fence system can contribute to errors. Lack of knowledge or negligence may lead to suboptimal performance or even safety hazards. Like when we set the module off position by chance we call off after saying this during testing we were saying off then it can detect our sound and never give us a response so these kinds of errors can also happen during this project.

It is crucial to address these errors and challenges by ensuring proper design, installation, and maintenance of the electric fence system. Regular monitoring, testing, and adjustments can help optimize its effectiveness in deterring locusts.

5.4. SDGs Mapping in Project

5.4.1. Establish Good Health And Well Being

This project is safe, pollution free and effective. As there is no by-product being made, so it's not harmful for human health.

5.4.2. Erase Hunger

It protects farms to be eaten or ruined by locusts. As a result, the dead locusts can be used as fertilizers that are good for plant health and growth.

As locusts can eat 38 persons' food at one time, so by using electric fencing technology to kill them and protect our farms from being ruined.

5.4.3. Create Decent Work and Economic Growth

The system is sustainable and economical for consumers.

5.5. Environment, Health and Safety

- EHS is an acronym for the set that studies and implements the practical aspects of protecting the environment and maintaining health and safety at occupation.
- This project is based on the protection of farms which means it is related to safety and health and as we are not using any harmful substance so, it also deals with the environment.

5.6. CEP Attributes

Attributes	Justification
Range of Conflicting requirements	Economical, Highly efficient , environment friendly and
Familiarity of issue	Famine, Starvation, Pollution(air, land, noise) and health issues
Interdependence	Engineers, Geographical Experts, farmers, Land Owners, Nutritionists, Historians, DVM's. they do not work independently.

5.7. CEA Attributes

Attributes	Justification
Innovation	Rather than using traditional methods we are implementing principal based approach to have a solution to protect food.
Consequences to society and the environment	System is completely pollution free as well as it is also giving us fertilizer in return in the form of locusts.
Familiarity	Unlike previous conventional methods like deafening sound, smoke and pesticides we are using electromagnetic field detection.

5.8. Conclusion

The development and implementation of an electric fence system to combat locust attacks in Pakistan hold significant promise for protecting agricultural lands, ensuring food security, and mitigating the negative impacts of conventional pest control methods. The electric fence system, designed to generate high-voltage pulses upon contact with locusts, acts as a deterrent, preventing insects from entering agricultural areas and causing extensive damage to crops. By adopting this sustainable operating system, Pakistan can address the longstanding challenge of locust infestations more effectively. The electric fences offer several advantages, including cost-effectiveness, ease of installation, and reduced risk of harm to livestock compared to traditional physical barriers and pesticide use. The system aligns with the global shift towards environmentally friendly and targeted pest control measures, minimizing the negative impacts on the environment and human health.

While ancient and traditional methods have been employed in the past, modern technologies and collaborations with international organizations like JICA and FAO are key to enhancing locust monitoring and control capacities. By strengthening monitoring networks and improving early warning systems, Pakistan can take proactive measures to prevent locust swarms and minimize their impact on agricultural lands. The successful implementation of the electric fence system requires ongoing research, development, and collaboration between government agencies, agricultural institutions, and international partners. Regular maintenance and monitoring of the fences, along with training programs for farmers and officials, will be essential for ensuring the system's long-term effectiveness.

Overall, the electric fence system represents a significant advancement in locust control strategies for Pakistan. By leveraging technology, sustainability, and international

cooperation, the country can protect its agricultural sector, enhance food security, and mitigate the devastating effects of locust swarms on its economy and environment.

5.9. Future Enhancement

Future enhancements in electric fencing against locusts can focus on several areas to further improve the effectiveness and efficiency of the system. Here are some potential areas of development:

5.9.1. Intelligent and adaptive systems

Developing intelligent and adaptive electric fence systems can optimize the use of resources and increase the system's efficacy. By integrating artificial intelligence and machine learning algorithms, the system can learn and adapt to locust behaviour patterns, optimize pulse frequency and intensity based on real-time conditions, and dynamically adjust the electric field configuration to target locusts more effectively.

5.9.2. Integrated pest management

Electric fencing can be combined with other pest control methods as part of an integrated pest management approach. For example, integrating the use of pheromone traps or bio pesticides within the electric fence system can attract and control locust populations more efficiently, reducing reliance on chemical pesticides.

5.9.3. Remote monitoring and control

Implementing remote monitoring and control capabilities in the electric fence system allows for real-time surveillance and management of multiple fence installations across a wide geographical area. This can be achieved through the use of IoT (Internet of Things) technologies, enabling farmers and authorities to monitor fence performance, receive alerts on locust activity, and remotely activate or adjust the electric field parameters.

5.9.4. Data analytics and predictive modeling

Leveraging data analytics and predictive modeling techniques can help in forecasting locust outbreaks, understanding locust behaviour patterns, and optimizing fence placement and configuration. By analyzing historical data, weather patterns, and other relevant factors, predictive models can provide insights and guidance for proactive locust management strategies.

5.9.5. Energy efficiency and sustainability

Research and development efforts can focus on optimizing the energy consumption of electric fence systems, exploring alternative power sources such as solar or wind, and

reducing the overall environmental footprint of the technology. Energy-efficient components and designs can contribute to the sustainability of the system.

5.9.6. Public Awareness and stakeholder engagement

Increasing public awareness about the benefits and effectiveness of electric fencing against locusts is crucial. Conducting awareness campaigns, providing training programs, and engaging with farmers, agricultural communities, and relevant stakeholders can ensure widespread adoption and support for this sustainable locust control method.

By continuously exploring and implementing these future enhancements, electric fencing against locusts can become even more efficient, environmentally friendly, and economically viable, contributing to the long-term protection of agricultural lands and the overall well-being of farming communities.

5.9.7. Magnetic Field

The major enhancement in this project is a conversion from an electric field to a magnetic field can be achieved by adding a magnetic coil or electromagnet to the fencing structure. These coils can be strategically placed to create a magnetic field around the fence. When an electric current passes through these coils, it creates a magnetic field that can interact with the locusts in various ways.

Magnetic fields can potentially affect locusts by disrupting their navigation or sensory systems. Locusts rely on various mechanisms, such as their ability to detect electric fields, to right themselves and find food sources. By introducing a magnetic field, it can interfere with their natural navigation abilities, making it difficult for them to reach or penetrate fences.

Appendix

```
#include <SoftwareSerial.h>
#include "VoiceRecognitionV3.h"

/**
 * Connection
 * Arduino    VoiceRecognitionModule
 * 2   ------->     TX
 * 3   ------->     RX
 */
VR myVR(10,11);    // 10:RX 11:TX for Arduino MEGA.

/**********************************************************************/
/** declare print functions */
void printSeperator();
void printSignature(uint8_t *buf, int len);
void printVR(uint8_t *buf);
void printLoad(uint8_t *buf, uint8_t len);
void printTrain(uint8_t *buf, uint8_t len);
void printCheckRecognizer(uint8_t *buf);
void printUserGroup(uint8_t *buf, int len);
void printCheckRecord(uint8_t *buf, int num);
void printCheckRecordAll(uint8_t *buf, int num);
void printSigTrain(uint8_t *buf, uint8_t len);
void printSystemSettings(uint8_t *buf, int len);
void printHelp(void);

/**********************************************************************/
// command analyze part
```

```
#define CMD_BUF_LEN      64+1
#define CMD_NUM     10
typedef int (*cmd_function_t)(int, int);
uint8_t cmd[CMD_BUF_LEN];
uint8_t cmd_cnt;
uint8_t *paraAddr;
int receiveCMD();
int checkCMD(int len);
int checkParaNum(int len);
int findPara(int len, int paraNum, uint8_t **addr);
int compareCMD(uint8_t *para1 , uint8_t *para2, int len);

int cmdTrain(int len, int paraNum);
int cmdLoad(int len, int paraNum);
int cmdTest(int len, int paraNum);
int cmdVR(int len, int paraNum);
int cmdClear(int len, int paraNum);
int cmdRecord(int len, int paraNum);
int cmdSigTrain(int len, int paraNum);
int cmdGetSig(int len, int paraNum);
int cmdSettings(int len, int paraNum);
int cmdHelp(int len, int paraNum);
/** cmdList, cmdLen, cmdFunction has correspondence */
const char cmdList[CMD_NUM][10] = {  // command list table
  {
    "train"  }
  ,
  {
    "load"  }
  ,
```

```
  {
    "clear"  }
  ,
  {
    "vr"  }
  ,
  {
    "record"  }
  ,
  {
    "sigtrain"  }
  ,
  {
    "getsig"  }
  ,
  {
    "Settings"  }
  ,
  {
    "test"  }
  ,
  {
    "help"  }
  ,
};
const char cmdLen[CMD_NUM]= {    // command length
  5,  //  {"train"},
  4,  //  {"load"},
  5,  //  {"clear"},
  2,  //  {"vr"},
```

```
  6,  //  {"record"},
  8,  //  {"sigtrain"},
  6,  //  {"getsig"},
  8,  //  {"Settings"},
  4,  //  {"test"},
  4,  //  {"help"}
};
cmd_function_t cmdFunction[CMD_NUM]={      // command handle fuction(function pointer
table)
  cmdTrain,
  cmdLoad,
  cmdClear,
  cmdVR,
  cmdRecord,
  cmdSigTrain,
  cmdGetSig,
  cmdSettings,
  cmdTest,
  cmdHelp,
};

/*******************************************************************************/
/** temprory data */
uint8_t buf[255];
uint8_t records[7]; // save record

void setup(void)
{
  myVR.begin(9600);
```

```
  /** initialize */
  Serial.begin(115200);
  Serial.println(F("Elechouse Voice Recognition V3 Module \"train\" sample."));

  printSeperator();
  Serial.println(F("Usage:"));
  printSeperator();
  printHelp();
  printSeperator();
  cmd_cnt = 0;
}

void loop(void)
{
  int len, paraNum, paraLen, i;

  /** receive Serial command */
  len = receiveCMD();
  if(len>0){
    /** check if the received command is valid */
    if(!checkCMD(len)){

      /** check parameter number of the received command  */
      paraNum = checkParaNum(len);

      /** display the receved command back */
      Serial.write(cmd, len);

      /** find the first parameter */
      paraLen = findPara(len, 1, &paraAddr);
```

```
  /** compare the received command with command in the list */
  for(i=0; i<CMD_NUM; i++){
    /** compare command length */
    if(paraLen == cmdLen[i]){
      /** compare command content */
      if( compareCMD(paraAddr, (uint8_t *)cmdList[i], paraLen) == 0 ){
        /** call command function */
        if( cmdFunction[i](len, paraNum) != 0){
          printSeperator();
          Serial.println(F("Command Format Error!"));
          printSeperator();
        }
        break;
      }
    }
  }

  /** command is not supported*/
  if(i == CMD_NUM){
    printSeperator();
    Serial.println(F("Unkonwn command"));
    printSeperator();
  }
}
else{
  /** received command is invalid */
  printSeperator();
  Serial.println(F("Command format error"));
  printSeperator();
```

```
    }
  }

  /** try to receive recognize result */
  int ret;
  ret = myVR.recognize(buf, 50);
  if(ret>0){
    /** voice recognized, print result */
    printVR(buf);
  }
}

/**
 * @brief   receive command from Serial.
 * @param   NONE.
 * @retval  command length, if no command receive return -1.
 */
int receiveCMD()
{
  int ret;
  int len;
  unsigned long start_millis;
  start_millis = millis();
  while(1){
    ret = Serial.read();
    if(ret>0){
      start_millis = millis();
      cmd[cmd_cnt] = ret;
      if(cmd[cmd_cnt] == '\n'){
        len = cmd_cnt+1;
```

```
      cmd_cnt = 0;
      return len;
     }
     cmd_cnt++;
     if(cmd_cnt == CMD_BUF_LEN){
      cmd_cnt = 0;
      return -1;
     }
    }

    if(millis() - start_millis > 100){
     cmd_cnt = 0;
     return -1;
    }
  }
}

/**
 * @brief   compare two commands, case insensitive.
 * @param   para1  -->  command buffer 1
 * para2  -->  command buffer 2
 * len    -->  buffer length
 * @retval  0  --> equal
 * -1  --> unequal
 */
int compareCMD(uint8_t *para1 , uint8_t *para2, int len)
{
  int i;
  uint8_t res;
  for(i=0; i<len; i++){
```

```
    res = para2[i] - para1[i];
    if(res != 0 && res != 0x20){
      res = para1[i] - para2[i];
      if(res != 0 && res != 0x20){
        return -1;
      }
    }
  }
  return 0;
}

/**
 * @brief   Check command format.
 * @param   len  -->  command length
 * @retval  0  -->  command is valid
 * -1  -->  command is invalid
 */
int checkCMD(int len)
{
  int i;
  for(i=0; i<len; i++){
    if(cmd[i] > 0x1F && cmd[i] < 0x7F){

    }
    else if(cmd[i] == '\t' || cmd[i] == ' ' || cmd[i] == '\r' || cmd[i] == '\n'){

    }
    else{
      return -1;
    }
```

```
  }
  return 0;
}

/**
 * @brief   Check the number of parameters in the command
 * @param   len  -->  command length
 * @retval  number of parameters
 */
int checkParaNum(int len)
{
  int cnt=0, i;
  for(i=0; i<len; ){
    if(cmd[i]!='\t' && cmd[i]!=' ' && cmd[i] != '\r' && cmd[i] != '\n'){
      cnt++;
      while(cmd[i] != '\t' && cmd[i] != ' ' && cmd[i] != '\r' && cmd[i] != '\n'){
        i++;
      }
    }
    i++;
  }
  return cnt;
}

/**
 * @brief   Find the specified parameter.
 * @param   len       -->  command length
 * paraIndex -->  parameter index
 * addr      -->  return value. position of the parameter
 * @retval  length of specified parameter
```

```
 */
int findPara(int len, int paraIndex, uint8_t **addr)
{
  int cnt=0, i, paraLen;
  uint8_t dt;
  for(i=0; i<len; ){
    dt = cmd[i];
    if(dt!='\t' && dt!=' '){
      cnt++;
      if(paraIndex == cnt){
        *addr = cmd+i;
        paraLen = 0;
        while(cmd[i] != '\t' && cmd[i] != ' ' && cmd[i] != '\r' && cmd[i] != '\n'){
          i++;
          paraLen++;
        }
        return paraLen;
      }
      else{
        while(cmd[i] != '\t' && cmd[i] != ' ' && cmd[i] != '\r' && cmd[i] != '\n'){
          i++;
        }
      }
    }
    else{
      i++;
    }
  }
  return -1;
}
```

```
int cmdHelp(int len, int paraNum)
{
  if(paraNum != 1){
    return -1;
  }
  printSeperator();
  printHelp();
  printSeperator();
  return 0;
}

/**
 * @brief   Handle "train" command
 * @param   len     --> command length
 * paraNum --> number of parameters
 * @retval  0 --> success
 * -1 --> Command format error
 */
int cmdTrain(int len, int paraNum)
{
  int i, ret;
  if(paraNum < 2 || paraNum > 8 ){
    return -1;
  }

  for(i=2; i<=paraNum; i++){
    findPara(len, i, &paraAddr);
    records[i-2] = atoi((char *)paraAddr);
    if(records[i-2] == 0 && *paraAddr != '0'){
```

```
    return -1;
   }
  }
  printSeperator();
  ret = myVR.train(records, paraNum-1, buf);
  //  ret = myVR.train(records, paraNum-1);
  if(ret >= 0){
   printTrain(buf, ret);
  }
  else if(ret == -1){
   Serial.println(F("Train failed."));
  }
  else if(ret == -2){
   Serial.println(F("Train Timeout."));
  }
  printSeperator();
  return 0;
}

/**
 * @brief   Handle "load" command
 * @param   len     --> command length
 * paraNum --> number of parameters
 * @retval  0 --> success
 * -1 --> Command format error
 */
int cmdLoad(int len, int paraNum)
{
  int i, ret;
  if(paraNum < 2 || paraNum > 8 ){
```

```
    return -1;
  }

  for(i=2; i<=paraNum; i++){
    findPara(len, i, &paraAddr);
    records[i-2] = atoi((char *)paraAddr);
    if(records[i-2] == 0 && *paraAddr != '0'){
      return -1;
    }
  }
  //  myVR.writehex(records, paraNum-1);
  ret = myVR.load(records, paraNum-1, buf);
  printSeperator();
  if(ret >= 0){
    printLoad(buf, ret);
  }
  else{
    Serial.println(F("Load failed or timeout."));
  }
  printSeperator();
  return 0;
}

/**
 * @brief   Handle "clear" command
 * @param   len     --> command length
 * paraNum --> number of parameters
 * @retval  0 --> success
 * -1 --> Command format error
 */
```

```
int cmdClear(int len, int paraNum)
{
 if(paraNum != 1){
  return -1;
 }
 if(myVR.clear() == 0){
  printSeperator();
  Serial.println(F("Recognizer cleared."));
  printSeperator();
 }
 else{
  printSeperator();
  Serial.println(F("Clear recognizer failed or timeout."));
  printSeperator();
 }
 return 0;
}

/**
 * @brief   Handle "vr" command
 * @param   len     --> command length
 * paraNum --> number of parameters
 * @retval  0 --> success
 * -1 --> Command format error
 */
int cmdVR(int len, int paraNum)
{
 int ret;
 if(paraNum != 1){
  return -1;
```

```
  }
  ret = myVR.checkRecognizer(buf);
  if(ret<=0){
    printSeperator();
    Serial.println(F("Check recognizer failed or timeout."));
    printSeperator();
    return 0;
  }
  printSeperator();
  printCheckRecognizer(buf);
  printSeperator();
  return 0;
}

/**
 * @brief   Handle "record" command
 * @param   len     --> command length
 * paraNum --> number of parameters
 * @retval  0 --> success
 * -1 --> Command format error
 */
int cmdRecord(int len, int paraNum)
{
  int ret;
  if(paraNum == 1){
    ret = myVR.checkRecord(buf);
    printSeperator();
    if(ret>=0){
      printCheckRecordAll(buf, ret);
    }
```

```
    else{
      Serial.println(F("Check record failed or timeout."));
    }
    printSeperator();
  }
  else if(paraNum < 9){
    for(int i=2; i<=paraNum; i++){
      findPara(len, i, &paraAddr);
      records[i-2] = atoi((char *)paraAddr);
      if(records[i-2] == 0 && *paraAddr != '0'){
        return -1;
      }
    }

    ret = myVR.checkRecord(buf, records, paraNum-1);    // auto clean duplicate records
    printSeperator();
    if(ret>=0){
      printCheckRecord(buf, ret);
    }
    else{
      Serial.println(F("Check record failed or timeout."));
    }
    printSeperator();
  }
  else{
    return -1;
  }
  return 0;
}
```

```
/**
 * @brief   Handle "sigtrain" command
 * @param   len     --> command length
 * paraNum --> number of parameters
 * @retval  0 --> success
 * -1 --> Command format error
 */
int cmdSigTrain(int len, int paraNum)
{
  int ret, sig_len;
  uint8_t *lastAddr;
  if(paraNum < 2){
    return -1;
  }

  findPara(len, 2, &paraAddr);
  records[0] = atoi((char *)paraAddr);
  if(records[0] == 0 && *paraAddr != '0'){
    return -1;
  }

  findPara(len, 3, &paraAddr);
  sig_len = findPara(len, paraNum, &lastAddr);
  sig_len +=( (unsigned int)lastAddr - (unsigned int)paraAddr );

  printSeperator();
  ret = myVR.trainWithSignature(records[0], paraAddr, sig_len, buf);
  //  ret = myVR.trainWithSignature(records, paraNum-1);
  if(ret >= 0){
    printSigTrain(buf, ret);
```

```
  }
  else{
    Serial.println(F("Train with signature failed or timeout."));
  }
  printSeperator();

  return 0;
}

/**
 * @brief   Handle "getsig" command
 * @param   len     --> command length
 * paraNum --> number of parameters
 * @retval  0 --> success
 * -1 --> Command format error
 */
int cmdGetSig(int len, int paraNum)
{
  int ret;
  if(paraNum != 2){
    return -1;
  }

  findPara(len, 2, &paraAddr);
  records[0] = atoi((char *)paraAddr);
  if(records[0] == 0 && *paraAddr != '0'){
    return -1;
  }

  ret = myVR.checkSignature(records[0], buf);
```

```
  printSeperator();
  if(ret == 0){
    Serial.println(F("Signature isn't set."));
  }
  else if(ret > 0){
    Serial.print(F("Signature:"));
    printSignature(buf, ret);
    Serial.println();
  }
  else{
    Serial.println(F("Get sig error or timeout."));
  }
  printSeperator();

  return 0;
}

/**
 * @brief   Handle "test" command
 * @param   len     --> command length
 * paraNum --> number of parameters
 * @retval  0 --> success
 * -1 --> Command format error
 */
int cmdTest(int len, int paraNum)
{
  printSeperator();
  Serial.println(F("TEST is not supported."));
  printSeperator();
```

```
  return 0;
}

int cmdSettings(int len, int paraNum)
{
  int ret;
  if(paraNum != 1){
    return -1;
  }
  ret = myVR.checkSystemSettings(buf);
  if( ret > 0){
    printSeperator();
    printSystemSettings(buf, ret);
    printSeperator();
  }
  else{
    printSeperator();
    Serial.println(F("Check system settings error or timeout"));
    printSeperator();
  }
  return 0;
}

/********************************************************************************/
/**
 * @brief   Print signature, if the character is invisible,
 * print hexible value instead.
 * @param   buf     --> command length
 * len     --> number of parameters
 */
```

```
void printSignature(uint8_t *buf, int len)
{
 int i;
 for(i=0; i<len; i++){
  if(buf[i]>0x19 && buf[i]<0x7F){
   Serial.write(buf[i]);
  }
  else{
   Serial.print(F("["));
   Serial.print(buf[i], HEX);
   Serial.print(F("]"));
  }
 }
}

/**
 * @brief   Print signature, if the character is invisible,
 * print hexible value instead.
 * @param   buf  -->  VR module return value when voice is recognized.
 * buf[0]  -->  Group mode(FF: None Group, 0x8n: User, 0x0n:System
 * buf[1]  -->  number of record which is recognized.
 * buf[2]  -->  Recognizer index(position) value of the recognized record.
 * buf[3]  -->  Signature length
 * buf[4]~buf[n] --> Signature
 */
void printVR(uint8_t *buf)
{
 Serial.println(F("VR Index\tGroup\tRecordNum\tSignature"));

 Serial.print(buf[2], DEC);
```

```
  Serial.print(F("\t\t"));

  if(buf[0] == 0xFF){
    Serial.print(F("NONE"));
  }
  else if(buf[0]&0x80){
    Serial.print(F("UG "));
    Serial.print(buf[0]&(~0x80), DEC);
  }
  else{
    Serial.print(F("SG "));
    Serial.print(buf[0], DEC);
  }
  Serial.print(F("\t"));

  Serial.print(buf[1], DEC);
  Serial.print(F("\t\t"));
  if(buf[3]>0){
    printSignature(buf+4, buf[3]);
  }
  else{
    Serial.print(F("NONE"));
  }
  Serial.println(F("\r\n"));
}

/**
 * @brief   Print seperator. Print 80 '-'.
 */
void printSeperator()
```

```
{
  for(int i=0; i<80; i++){
    Serial.write('-');
  }
  Serial.println();
}

/**
 * @brief   Print recoginizer status.
 * @param   buf  -->  VR module return value when voice is recognized.
 * buf[0]     -->  Number of valid voice records in recognizer
 * buf[i+1]   -->  Record number.(0xFF: Not loaded(Nongroup mode), or not set (Group mode))
(i= 0, 1, ... 6)
 * buf[8]     -->  Number of all voice records in recognizer
 * buf[9]     -->  Valid records position indicate.
 * buf[10]    -->  Group mode indicate(FF: None Group, 0x8n: User, 0x0n:System)
 */
void printCheckRecognizer(uint8_t *buf)
{
  Serial.print(F("All voice records in recognizer: "));
  Serial.println(buf[8], DEC);
  Serial.print(F("Valid voice records in recognizer: "));
  Serial.println(buf[0], DEC);
  if(buf[10] == 0xFF){
    Serial.println(F("VR is not in group mode."));
  }
  else if(buf[10]&0x80){
    Serial.print(F("VR is in user group mode:"));
    Serial.println(buf[10]&0x7F, DEC);
  }
```

```
  else{
    Serial.print(F("VR is in system group mode:"));
    Serial.println(buf[10], DEC);
  }
  Serial.println(F("VR Index\tRecord\t\tComment"));
  for(int i=0; i<7; i++){
    Serial.print(i, DEC);
    Serial.print(F("\t\t"));
    if(buf[i+1] == 0xFF){
      if(buf[10] == 0xFF){
        Serial.print(F("Unloaded\tNONE"));
      }
      else{
        Serial.print(F("Not Set\t\tNONE"));
      }
    }
    else{
      Serial.print(buf[i+1], DEC);
      Serial.print(F("\t\t"));
      if(buf[9]&(1<<i)){
        Serial.print(F("Valid"));
      }
      else{
        Serial.print(F("Untrained"));
      }
    }

    Serial.println();
  }
}
```

```
/**
 * @brief   Print record train status.
 * @param   buf  -->  Check record command return value
 * buf[0]     -->  Number of checked records
 * buf[2i+1]  -->  Record number.
 * buf[2i+2]  -->  Record train status. (00: untrained, 01: trained, FF: record value out of range)
 * (i = 0 ~ buf[0]-1 )
 * num  -->  Number of trained records
 */
void printCheckRecord(uint8_t *buf, int num)
{
  Serial.print(F("Check "));
  Serial.print(buf[0], DEC);
  Serial.println(F(" records."));

  Serial.print(num, DEC);
  if(num>1){
    Serial.println(F(" records trained."));
  }
  else{
    Serial.println(F(" record trained."));
  }

  for(int i=0; i<buf[0]*2; i += 2){
    Serial.print(buf[i+1], DEC);
    Serial.print(F("\t-->\t"));
    switch(buf[i+2]){
    case 0x01:
      Serial.print(F("Trained"));
```

```
      break;
    case 0x00:
      Serial.print(F("Untrained"));
      break;
    case 0xFF:
      Serial.print(F("Record value out of range"));
      break;
    default:
      Serial.print(F("Unknown Stauts"));
      break;
    }
    Serial.println();
  }
}

/**
 * @brief   Print record train status.
 * @param   buf  -->  Check record command return value
 * buf[0]     -->  Number of checked records
 * buf[2i+1]  -->  Record number.
 * buf[2i+2]  -->  Record train status. (00: untrained, 01: trained, FF: record value out of range)
 * (i = 0 ~ buf[0]-1 )
 * num  -->  Number of trained records
 */
void printCheckRecordAll(uint8_t *buf, int num)
{
  Serial.print(F("Check 255"));
  Serial.println(F(" records."));

  Serial.print(num, DEC);
```

```
  if(num>1){
    Serial.println(F(" records trained."));
  }
  else{
    Serial.println(F(" record trained."));
  }
  myVR.writehex(buf, 255);
  for(int i=0; i<255; i++){
    if(buf[i] == 0xF0){
      continue;
    }
    Serial.print(i, DEC);
    Serial.print(F("\t-->\t"));
    switch(buf[i]){
    case 0x01:
      Serial.print(F("Trained"));
      break;
    case 0x00:
      Serial.print(F("Untrained"));
      break;
    case 0xFF:
      Serial.print(F("Record value out of range"));
      break;
    default:
      Serial.print(F("Unknown Stauts"));
      break;
    }
    Serial.println();
  }
}
```

```
/**
 * @brief   Print check user group result.
 * @param   buf  -->  Check record command return value
 * buf[8i]    -->  group number.
 * buf[8i+1]  -->  group position 0 status.
 * buf[8i+2]  -->  group position 1 status.
 * ...               ...
 * buf[8i+6]  -->  group position 5 status.
 * buf[8i+7]  -->  group position 6 status.
 * (i = 0 ~ len)
 * len  -->  number of checked groups
 */
void printUserGroup(uint8_t *buf, int len)
{
  int i, j;
  Serial.println(F("Check User Group:"));
  for(i=0; i<len; i++){
    Serial.print(F("Group:"));
    Serial.println(buf[8*i]);
    for(j=0; j<7; j++){
      if(buf[8*i+1+j] == 0xFF){
        Serial.print(F("NONE\t"));
      }
      else{
        Serial.print(buf[8*i+1+j], DEC);
        Serial.print(F("\t"));
      }
    }
    Serial.println();
```

```
  }
}

/**
 * @brief   Print "load" command return value.
 * @param   buf  -->  "load" command return value
 * buf[0]    -->  number of records which are load successfully.
 * buf[2i+1]  -->  record number
 * buf[2i+2]  -->  record load status.
 * 00 --> Loaded
 * FC --> Record already in recognizer
 * FD --> Recognizer full
 * FE --> Record untrained
 * FF --> Value out of range"
 * (i = 0 ~ (len-1)/2 )
 * len  -->  length of buf
 */
void printLoad(uint8_t *buf, uint8_t len)
{
  if(len == 0){
    Serial.println(F("Load Successfully."));
    return;
  }
  else{
    Serial.print(F("Load success: "));
    Serial.println(buf[0], DEC);
  }
  for(int i=0; i<len-1; i += 2){
    Serial.print(F("Record "));
    Serial.print(buf[i+1], DEC);
```

```
    Serial.print(F("\t"));
    switch(buf[i+2]){
    case 0:
      Serial.println(F("Loaded"));
      break;
    case 0xFC:
      Serial.println(F("Record already in recognizer"));
      break;
    case 0xFD:
      Serial.println(F("Recognizer full"));
      break;
    case 0xFE:
      Serial.println(F("Record untrained"));
      break;
    case 0xFF:
      Serial.println(F("Value out of range"));
      break;
    default:
      Serial.println(F("Unknown status"));
      break;
    }
  }
}

/**
 * @brief   Print "train" command return value.
 * @param   buf  -->  "train" command return value
 * buf[0]    -->  number of records which are trained successfully.
 * buf[2i+1]  -->  record number
 * buf[2i+2]  -->  record train status.
```

```
 * 00 --> Trained
 * FE --> Train Time Out
 * FF --> Value out of range"
 * (i = 0 ~ len-1 )
 * len  -->  length of buf
 */
void printTrain(uint8_t *buf, uint8_t len)
{
  if(len == 0){
    Serial.println(F("Train Finish."));
    return;
  }
  else{
    Serial.print(F("Train success: "));
    Serial.println(buf[0], DEC);
  }
  for(int i=0; i<len-1; i += 2){
    Serial.print(F("Record "));
    Serial.print(buf[i+1], DEC);
    Serial.print(F("\t"));
    switch(buf[i+2]){
    case 0:
      Serial.println(F("Trained"));
      break;
    case 0xFE:
      Serial.println(F("Train Time Out"));
      break;
    case 0xFF:
      Serial.println(F("Value out of range"));
      break;
```

```
    default:
      Serial.print(F("Unknown status "));
      Serial.println(buf[i+2], HEX);
      break;
    }
  }
}

/**
 * @brief   Print "sigtrain" command return value.
 * @param   buf  -->  "sigtrain" command return value
 * buf[0]  -->  number of records which are trained successfully.
 * buf[1]  -->  record number
 * buf[2]  -->  record train status.
 * 00 --> Trained
 * F0 --> Trained, signature truncate
 * FE --> Train Time Out
 * FF --> Value out of range"
 * buf[3] ~ buf[len-1] --> Signature.
 * len  -->  length of buf
 */
void printSigTrain(uint8_t *buf, uint8_t len)
{
  if(len == 0){
    Serial.println(F("Train With Signature Finish."));
    return;
  }
  else{
    Serial.print(F("Success: "));
    Serial.println(buf[0], DEC);
```

```
  }
  Serial.print(F("Record "));
  Serial.print(buf[1], DEC);
  Serial.print(F("\t"));
  switch(buf[2]){
  case 0:
    Serial.println(F("Trained"));
    break;
  case 0xF0:
    Serial.println(F("Trained, signature truncate"));
    break;
  case 0xFE:
    Serial.println(F("Train Time Out"));
    break;
  case 0xFF:
    Serial.println(F("Value out of range"));
    break;
  default:
    Serial.print(F("Unknown status "));
    Serial.println(buf[2], HEX);
    break;
  }
  Serial.print(F("SIG: "));
  Serial.write(buf+3, len-3);
  Serial.println();
}

/**
 * @brief   Print "settings" command return value.
 * @param   buf  -->  "settings" command return value
```

```
 * buf[0]  -->  number of records which are trained successfully.
 * buf[1]  -->  record number
 * buf[2]  -->  record train status.
 * 00 --> Trained
 * F0 --> Trained, signature truncate
 * FE --> Train Time Out
 * FF --> Value out of range"
 * buf[3] ~ buf[len-1] --> Signature.
 * len  -->  length of buf
 */

const unsigned int io_pw_tab[16]={
  10,  15,  20,  25,  30,  35,  40,  45,
  50,  75,  100, 200, 300, 400, 500, 1000
};

void printSystemSettings(uint8_t *buf, int len)
{

  switch(buf[0]){
  case 0:
  case 3:
    Serial.println(F("Baud rate: 9600"));
    break;
  case 1:
    Serial.println(F("Baud rate: 2400"));
    break;
  case 2:
    Serial.println(F("Baud rate: 4800"));
    break;
```

```
case 4:
  Serial.println(F("Baud rate: 19200"));
  break;
case 5:
  Serial.println(F("Baud rate: 38400"));
  break;
default:
  Serial.println(F("Baud rate: UNKONOWN"));
  break;
}

switch(buf[1]){
case 0:
case 0xFF:
  Serial.println(F("Outpu IO Mode: Pulse"));
  break;
case 1:
  Serial.println(F("Outpu IO Mode: Toggle"));
  break;
case 2:
  Serial.println(F("Outpu IO Mode: Clear(When recognized) "));
  break;
case 3:
  Serial.println(F("Outpu IO Mode: Set(When recognized)"));
  break;
default:
  Serial.println(F("Output IO Mode: UNKONOWN"));
  break;
}
```

```
if(buf[2] > 15){
  Serial.println(F("Pulse width: UNKONOWN"));
}
else{
  Serial.print(F("Pulse Width: "));
  Serial.print(io_pw_tab[buf[2]], DEC);
  Serial.println(F("ms"));
}

if(buf[3] == 0 || buf[3] == 0xFF){
  Serial.println(F("Auto Load: disable"));
}
else{
  Serial.println(F("Auto Load: enable"));
}

switch(buf[4]){
case 0:
case 0xFF:
  Serial.println(F("Group control by external IO: disabled"));
  break;
case 1:
  Serial.println(F("Group control by external IO: system group selected"));
  break;
case 2:
  Serial.println(F("Group control by external IO: user group selected"));
  break;
default:
  Serial.println(F("Group control by external IO: UNKNOWN STATUS"));
  break;
```

```
  }
}


void printHelp(void)
{
  Serial.println(F("COMMAND       FORMAT                    EXAMPLE                                  Comment"));
  printSeperator();
  //  Serial.println(F("-------------------------------------------------------------------------------------------------------"));
  Serial.println(F("train         train (r0) (r1)...        train 0 2 45             Train records"));
  Serial.println(F("load          load (r0) (r1) ...        load 0 51 2 3            Load records"));
  Serial.println(F("clear         clear                     clear                    remove all records in Recognizer"));
  Serial.println(F("record        record / record (r0) (r1)...  record / record 0 79      Check record train status"));
  Serial.println(F("vr            vr                        vr                       Check recognizer status"));
  Serial.println(F("getsig        getsig (r)                getsig 0                 Get signature of record (r)"));
  Serial.println(F("sigtrain      sigtrain (r) (sig)        sigtrain 0 ZERO          Train one record(r) with signature(sig)"));
  Serial.println(F("settings      settings                  settings                 Check current system settings"));
  Serial.println(F("help          help                      help                     print this message"));
}
```

References

1. "The locust crisis: The World Bank's response - World". ReliefWeb. Retrieved 2020-04-28.
2. "Nuvem de gafanhotos na Argentina deixa fronteira com Brasil em alerta". R7.com (in Brazilian Portuguese). 2020-06-23. Retrieved 2020-06-24
3. "Alarm as coronavirus curbs disrupt East Africa fight on locusts". www.aljazeera.com. Retrieved 2020-04-28.
4. "Desert Locust Bulletin - General situation during October 2020" (PDF). Food and Agriculture Organization. 2 November 2020. Retrieved 7 November 2020.
5. "The Terrifying Science Behind the Locust Plagues of Africa". Wired. ISSN 1059-1028. Retrieved 2020-04-14.
6. Anstey, Michael L.; Rogers, Stephen M.; Ott, Swidbert R.; Burrows, Malcolm; Simpson, Stephen J. (2009-01-30). "Serotonin Mediates Behavioral Gregarization Underlying Swarm Formation in Desert Locusts". Science. **323** (5914): 627–630. Bibcode:2009Sci...323..627A. doi:10.1126/science.1165939. ISSN 0036-8075. PMID 19179529. S2CID 5448884.
7. "Frequently Asked Questions (FAQs) about locusts". www.fao.org. Retrieved 2020-11-0
8. "A plague of locusts has descended on East Africa. Climate change may be to blame". Science. 2020-02-14. Retrieved 2020-05-01.
9. ^ "Locust crisis poses a danger to millions, forecasters warn". the Guardian. 2020-03-20. Retrieved 2020-04-28
10. "A plague of locusts has descended on East Africa. Climate change may be to blame". Science. 2020-02-14. Retrieved 2020-04-28.
11. Ahmed, Kaamil (2020-03-20). "Locust crisis poses a danger to millions, forecasters warn". The Guardian. ISSN 0261-3077. Retrieved 2020-05-01
12. "Kenya Situation Report, 14 May 2020" (PDF). ReliefWeb. United Nations Office for the Coordination of Humanitarian Affairs. 14 May 2020. Retrieved 7 November 2020
13. "Food security fears as locusts destroy crops in Pakistan's worst plague since the 1990s". The Telegraph. 18 Nov 2019
14. "As swarms of desert locust descend upon Karachi, ministry says species pose no risk to food supply". Dawn. 11 November 2019.
15. Pakistan declares national emergency over locust swarms". Deutsche Welle. 2 Feb 2020
16. Dev Ankur Wadhawan (January 20, 2020). "Locusts' attack in western Rajasthan leaves farmers high and dry, ruin lakhs of hectares of crops". India Today. Retrieved 2020-05-04
17. Ahmed, A. 2019. Dawn News. As swarms of desert locust descend upon Karachi, ministry says species pose no risk to food supply, https://www.dawn.com/news/amp/1516067. Accessed: 15 March 2021.
18. Ali, M. 2020. Emergency declared in nine districts after Dera locust attack. Dawn News. https://reliefweb.int/report/pakistan/emergency-declared-nine-districts-after-dera-locust-attack. Accessed: 7 March 2021.
19. "Manual on safety" (PDF). Archived from the original (PDF) on 20 July 2011. Retrieved 19 April 2010.
20. E. Germany Builds Electric Fence", The Times, 28 March 1984
21. FAO, food chain crisis, FAO , http://www.fao.org/food-chain-crisis/home/en/ (2020), Accessed 3rd Sep 2020

22. Desert Locust Management : a Time for Change (English), World Bank Gr (1995), (accessed March 6, 2020), http://documents.worldbank.org/curated/en/298411468760511318/Desert-locust-management-a-time-for-change.
23. News, JICA, https://www.jica.go.jp/english/news/field/2021/20210603_01.html, June 3, 2021.
24. ^ "The electromechanical relay of Joseph Henry". Georgi Dalakov. Archived from the original on 2012-06-18. Retrieved 2012-06-21.
25. US 1037492, Kettering, Charles F., "Ignition system", published 2 November 1910, issued 3 September 1912
26. ^ US 1754265, Coursey, Philip Ray, "Electrical Condenser", published 23 June 1926, issued 15 April 1930
27. ^ Jump up to:a b "When was the SMPS power supply invented?". electronicspoint.com.
28. ^ "Electrical condensers (Open Library)". openlibrary.org.
29. "Arduino UNO for beginners - Projects, Programming and Parts". makerspaces.com. 7 February 2017. Retrieved 4 February 2018.
30. ^ "Arduino FAQ". 5 April 2013. Archived from the original on 27 November 2020. Retrieved 21 February 2018.
31. ^ Jump up to:a b "What is Arduino?". learn.sparkfun.com. Retrieved 4 February 2018.
32. https://mysite.du.edu/~jcalvert/tel/morse/morse.htm#
33. Tutorials, electronics Arduino Circuits, https://electronoobs.com/eng_arduino_tut165.php, December 26, 2021.