FINAL YEAR PROJECT REPORT

REAL-TIME PEST DETECTION AND FUMIGATION USING MACHINE LEARNING

In fulfillment of the requirement

for degree of

BEE (Electrical Engineering)

By

BURHANUDDIN ZOHAIR	**57310**	**BEE (ELECTRICAL)**
NOOR UL AIN AYAZ	**54187**	**BEE (ELECTRICAL)**
SYEDA BRIHA HAIDER	**57080**	**BEE (ELECTRICAL)**

SUPERVISED

BY

DR. MUKESH KUMAR MAHESHWARI

BAHRIA UNIVERSITY (KARACHI CAMPUS)

2018 - 2022

REAL-TIME PEST DETECTION AND FUMIGATION USING MACHINE LEARNING.

NOOR UL AIN AYAZ
SYEDA BRIHA HAIDER
BURHANUDDIN ZOHAIR

A project report submitted in partial fulfilment of the requirements for the award of the degree of Bachelor of Electrical (Electronic) Engineering

Electrical Engineering Bahria University, Karachi Campus

2022

DECLARATION

We hereby declare that this project report is based on our original work except for citations and quotations which have been duly acknowledged. We also declare that it has not been previously and concurrently submitted for any other degree or award at Bahria University or other institutions.

Signature : ___________________________

Name : Noor Ul Ain Ayaz

Reg No. 54187

Signature : ___________________________

Name : Syeda Briha Haider

Reg No. 57080

Signature : ___________________________

Name : Burhanuddin Zohair

Reg No. 57310

Date :

APPROVAL FOR SUBMISSION

We certify that this project report entitled **"REAL-TIME PEST DETECTION AND FUMIGATION USING MACHINE LEARNING**" was prepared by **NOOR UL AIN AYAZ ,SYEDA BRIHA HAIDER** and **BURHANUDDIN ZOHAIR** has met the required standard for submission in partial fulfilment of the requirements for the award of Bachelor of **Electrical (Electronic) Engineering** at Bahria University.

Approved by, **Dr. Mukesh Kumar Maheshwari**

Signature : ___________________________

Supervisor : **Dr. Mukesh Kumar Maheshwari**

Date : ___________________________

ACKNOWLEDGEMENTS

We would like to thank everyone who had contributed to the successful completion of this project. We would like to express our gratitude to our research supervisor, **Dr. Mukesh Kumar Maheshwari** for his invaluable advice, guidance and his enormous patience throughout the development of the research.

In addition, we would also like to express our gratitude to our loving parent and friends who had helped and given us encouragement.

REAL-TIME PEST DETECTION AND FUMIGATION USING MACHINE LEARNING.

ABSTRACT

Pakistan generates 70% economy from agricultural sector. It is the largest growing sector in Pakistan. It contributes 24% to the GDP. The geographical location and environmental conditions of Pakistan are suitable for the growth of most of the crops. Every crop requires different climatic condition to grow. Crops such as rice, cotton , vegetables and fruits are mainly exported to Europe and other countries. These countries face harsh climate which does not satisfy the necessary climatic conditions required for several crops and therefore these countries export these crops from countries like Pakistan due to its excellent crop quality , production and growth rate. The crop demand is directly dependant on the crop quality which is a major factor that must not be compromised. Providing the necessary conditions to grow and protecting from any diseases is the major goal that needs to be achieved. These crops are prone to get affected by any diseases that can be caused by different pests. Pests largely compromise the crop quality and destroy several crops which affects the economy. Therefore a pesticide needs to be used in order to get rid of any pests present on the crops that might compromise the crop quality. Large amounts of plants may contain several pests of different sizes and colors which may not be visible to human eye. Therefore, we have designed a robotic arm that can move around the crop field and effectively scan the plants for the presence of any pests on the crops and spraying pesticide to get rid of them. The pesticide used will get rid of these pests without causing any harm to the nature as it is environmental friendly. Using Innovative technology we will try to reduce wastage of crops, improve crop quality and reducing poverty. This will reduce labour work and promote increased crop demand resulting in good impact on the economy .

Table of Contents

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1
INTRODUCTION

1.1 Background

Agriculture is the largest and fastest growing sector in Pakistan. It is a large contributor to the economy of Pakistan ,contributes about 24% to the (Gross Domestic Product) GDP. Large part of the population is depended on Agricultural sector. It provides employment to the labors, food for the population as well as means of foreign exchange earning. The geographical location of Pakistan makes it suitable for the growth of most of the crops and therefore other countries living in harsh environments need to import these crops from countries like Pakistan.

The demand is directly related to the quality of these crops grown. The quality of the crops should be remarkable in order to export these crops. Several factors are responsible for compromising the crop quality. One major factor is the issue of Pests. Pests are creatures that can potentially destroy and damage the crops which would compromise the quality and therefore resulting in the crops not being exported. This results in huge amount of crops being wasted and affecting the economy.

To counter this problem, Computer vision and Machine learning can be used to locate and identify multiples pests present on the crops and getting rid of them by fumigating the plant using pesticide.

The idea behind the project is to create a robotic arm that can move in vertical farms to scan any pests present that may not be visible to human eye and get rid of them by spraying pesticide.

1.2 Literature Review

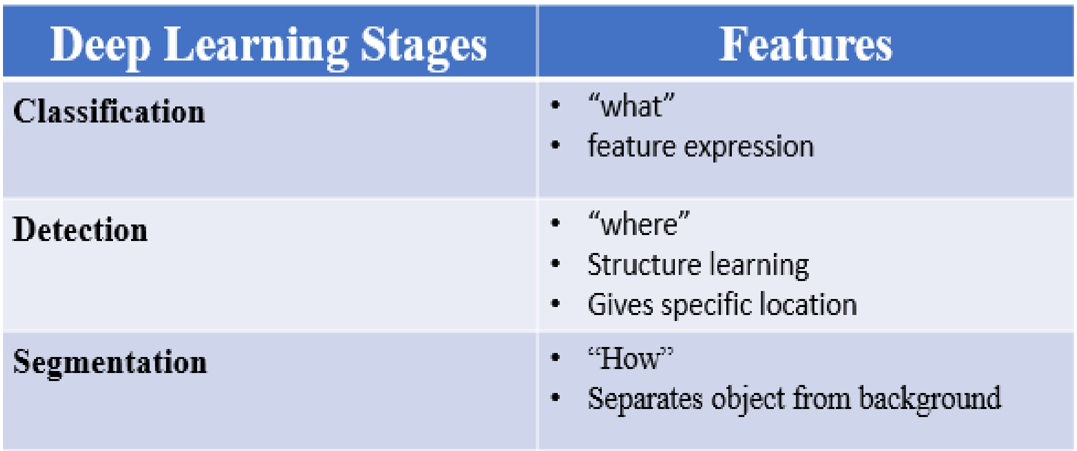
Computer Vision is a field of Artificial Intelligence that helps computers to understand information from images. It involves techniques to mimic human vision, by extracting information from images and understanding them [1] .It uses Machine Learning techniques to visualize the world.

Often Computer Vision and Image processing are used interchangeably.

Computer Vision gives computers the ability to understand digital world such as object detection , object recognition, image classification etc. Whereas Image Processing transforms/manipulates images from one form to another by filtering, enhancement , sharpening and restoration.
Both techniques are used widely and several work has been done in this field. These techniques have been used widely for object detection.

Machine Learning algorithms can be divided into supervised, unsupervised and semi-supervised learning , reinforcement learning, ensemble learning, instance based learning , multi-task learning and neural Network. as discussed in a study done on machine learning algorithms [3]. Our point of interest is neural Network as neural networks are algorithms designed to mimic human brain. These algorithms recognize relationship between some data , mimicking human brain. They give the best result and have become extremely popular. Neural networks can be of various types namely: supervised neural network ,unsupervised neural network and Reinforced Neural Network. Our aim is to perform object detection on insects for which we will use supervised neural network in which the model is trained on labelled data.

[6] Liu, Jun, and Xuewei Wang which is a research on Plant diseases and Pest's detection based on Deep learning. The Research paper identified a few researches. It mentions how image processing is a very common procedure used for plant diseases and pest detection but due to its limitations and a lot of disturbances caused while collection of images , this method tends to fail or give inaccurate results. In modern days, deep learning algorithms such as CNN are now being used in several applications. Doing pest detection using this modern deep learning method has proven very important and is very useful in several applications yielding great results. It stated some plant disease and pest detection methods. It further discussed deep learning stages as image classification ,segmentation and detection. Classification gives the information of an image such as its label Detection gives the location of an image . Segmentation separates the area of interest from the background. It further discusses classification networks such as Convolution Neural Network (CNN) and how it works.



Deep Learning Stages	Features
Classification	• “what” • feature expression
Detection	• “where” • Structure learning • Gives specific location
Segmentation	• “How” • Separates object from background

Figure 1: Deep Learning Stages

Research paper by T. Kasinathan, et al [15] is a study regarding the Classification and detection of insects in crops using machine learning techniques. This study applied different machine learning techniques such as artificial neural networks (ANN), support vector machine (SVM), k-nearest neighbors (KNN), naive bayes (NB) and convolutional neural network (CNN) to Wang , Xie , Dang, and IP102 datasets and compared the accuracy of each technique. Machine learning is a great and widely used technique for detection and classification of objects. These models were trained on the dataset to compare which model performs best. It was suggested that CNN is the most accurate technique compared to the rest as it gives 91.5% accuracy. This is a good accuracy also ensuring computation time is less .

Convolution Neural Network is a widely used deep learning algorithm useful in many applications. It works on the principle of convolution and has multiple layers in its structure [22]. CNN has the capability of recognizing varying patterns and is a collection of neurons within each layer which allows implicit learning. It has four layers each designated to perform different tasks. The layers are as follows Convolution Layer, Pooling Layer, Fully Connected Layer and Loss Layer. There are different architectures of CNN such as LeNet, AlexNet,GoogleNet , ResNet , MobileNet etc.

Research performed a comparative study of the performance of CNN architectures namely Resnet,Xception , shuffleNet,MobileNet and trained for tomato diseases dataset. Results showed that DenseNet gave the most accuracy but the number of parameters were the most. On the other hand, ShuffleNet and Mobilenet had the least number of parameters with MobileNet giving the accuracy like to GoogleNet.

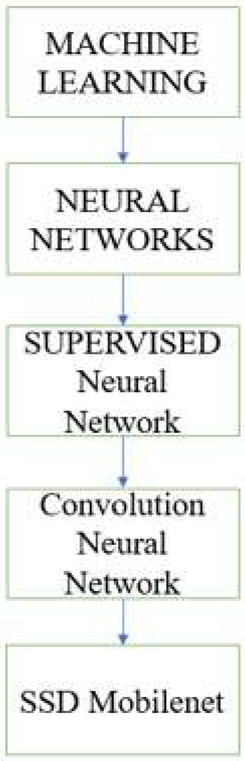


Figure 2:Neural network and its architecture

1.3 Problem Statements

Plant diseases is a major problem faced by farmers that directly affects their health and growth. Large amount of crops / plants are damaged due to the diseases which greatly impacts the economy. These diseases are caused by several factors but the major factor is the issue of pests. The pests tend to eat away and damage the plants. This is a problem which needs to be dealt with by getting rid of these pests present on plants.

1.4 Aims and Objectives

The following are the aims and objectives of our project :

- **Secure Plant health**

  Quality of the plant is the major factor in agricultural sector of Pakistan as it directly proportional to the Pakistan’s economy. So we are aimed to protect the plants (crops) from pests as they degrades the quality of the plant.

- **Reduce labor work**

  Reduction of human work load is another objective so that the robotic arm with proper pest killing system will be designed. The robot will perform the task of detection as well as killing the pest to reduce human work.

- **Save time**

  As compared to human , machine will perform the task in lesser amount of time.

- **Safe and limited usage of pesticides**

  Limited amount of pesticides will be sprayed to that particular area only where the pest will be detected.

- **Regular monitoring of plants**

  The robotic arm will keep monitoring the plants whenever desired without being dependent on humans.

1.5 Scope of Project

The purpose of our project is to develop a robotic arm that moves around fields in order to get rid of any pests present on the plants and ensure good plant health and quality. It uses LFR operation to move around fields. It is best suited to work in vertical farms. Nowadays, vertical farming has become extremely common and several cities due to the great amount of advantages that it has.

The robotic arm will scan every plant and get rid of any pest present on them by using a pesticide which is safe for the plants and the environment. It is designed in a way to scan plant from all angles. It will monitor the plants efficiently , reduce labor work and even detect pests that may be visible to human eye.

1.6 Sustainable Development Goals of Project

1.6.1 Introduction

There are 17 sustainable development goals (SDGs). These goals are designed to help take world better place. Bahria University Electrical Engineering department aims to contribute towards directing our Final Year Projects (FYPs) to be ***mapped with at least one of the SDGs*** so that our students play a direct role in the wellbeing of the world.

The Sustainable Development Goals (SDGs) are set to end poverty, protect the planet, and ensure that all people enjoy peace and prosperity.

The 17 SDGs are integrated and recognize that action in one area will affect outcomes in others, and that development must balance socio-economic and environmental sustainability.

Your project must be mapped with at least one of the below mentioned SDGs and you will provide justification that how your project is mapped with respective SDG.

1.6.2 Justification

The project maps on 2 SDG’s. Goal 1 states no Poverty. Our project will improve the quality of the plants by getting rid of any pests present on it , it will reduce wastage and increase growth and production thereby fulfilling the demand and decreasing poverty. Our Project uses new technology and therefore satisfies Goal 9 of the SDG’s learning outcomes.

1.6.3 Mapping of Sustainable Development Goals

SDGs learning Outcomes	**Temperature and Mask Scan Entry System for Covid Prevention**	
	Mapping	**SDG attainment Detail**
GOAL 1: No Poverty		
GOAL 2: Zero Hunger		
GOAL 3: Good Health and Well-being	✓	Minimizes the spread of Covid-19.
GOAL 4: Quality Education		
GOAL 5: Gender Equality		
GOAL 6: Clean Water and Sanitation		
GOAL 7: Affordable and Clean Energy		
GOAL 8: Decent Work and Economic Growth		
GOAL 9: Industry, Innovation and Infrastructure	✓	Uses New Technology
GOAL 10: Reduced Inequality		
GOAL 11: Sustainable Cities and Communities		
GOAL 12: Responsible Consumption and Production		
GOAL 13: Climate Action		
GOAL 14: Life Below Water		
GOAL 15: Life on Land		
GOAL 16: Peace and Justice Strong Institutions		
GOAL 17: Partnerships to achieve the Goal		

1.7 Environment Aspects of Project

1.7.1 Introduction

This project is designed to give you a field experience that will expose you to scientific principles of field work in environmental analysis and other activities related to the preparation of an environmental impact statement. The Project follows the Environmental Impact assessment.

1.7.2 Environmental Impact Assessment (EIA)

The idea behind the project is to safely get rid of the pests present on the plants. This will be done by using a pesticide. Our design uses a pesticide which is not harmful to the environment , it is environmental friendly unlike most which cause issues like ozone depletion. The pesticide is chemical free so it does not cause the crops to become toxic and makes them safe for human consumption. Our aim here is to keep the environment safe and secure by not using any such pollutants. The pesticide is 100% environmental friendly and will not cause any sort of pollution.

1.7.3 Environmental Impact Statement (EIS)

The project is environmental friendly and poses no harm to environment and to those consuming these crops.

CHAPTER 2

DESIGN AND METHODOLOGY

2.1 DESIGN

Our project is the combination of hardware and software. A device called jetson nano has been used which is a mini computer that will be trained on the created dataset of pests to detect the pest presence and a hardware model of a robotic arm will be designed to work in conjunction with the nano. The software and hardware will work simultaneously. Using the concepts of LFR the robotic arm will move around the field and scan for pests.

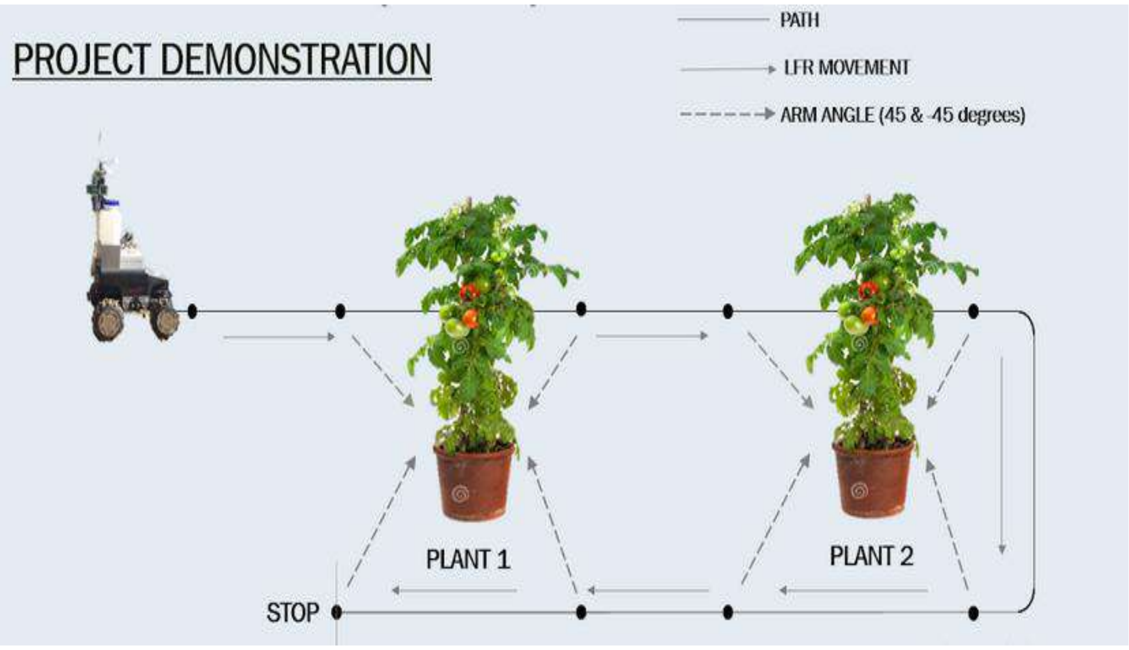


Figure 3:Demonstration

2.2 METHODOLOGY

Applying the concepts of AI and machine learning, the Jetson Nano has been trained to use pre-trained networks.It uses transfer learning which is a method of training pretrained network by transferring knowledge of previously trained network to new task. Neural networks are a type of machine learning technique that help mimic like human brain in order to perform object detection , recognition and segmentation. Installing necessary libraries and packages will enable the device to be trained on our dataset. . Once the device is trained it can detect the presence of the pests. To make the project movable LFR principle is used which will enable the robotic arm to follow a certain path. Horizontal , vertical and rotational movements are adjusted every time a plant is nearby which will be detected using ultrasonic sensors. Using Jetson Nano and Arduino as main controllers in conjunction with dc motors, servo motors IR sensors , rotatory encoders and Logitech c270 webcam as main components of the design. All the plants are scanned and once the presence of pests is detected motor is activated to spray pesticide on the pests to get rid of them. The arm keeps moving until all plants are scanned and pests are killed.

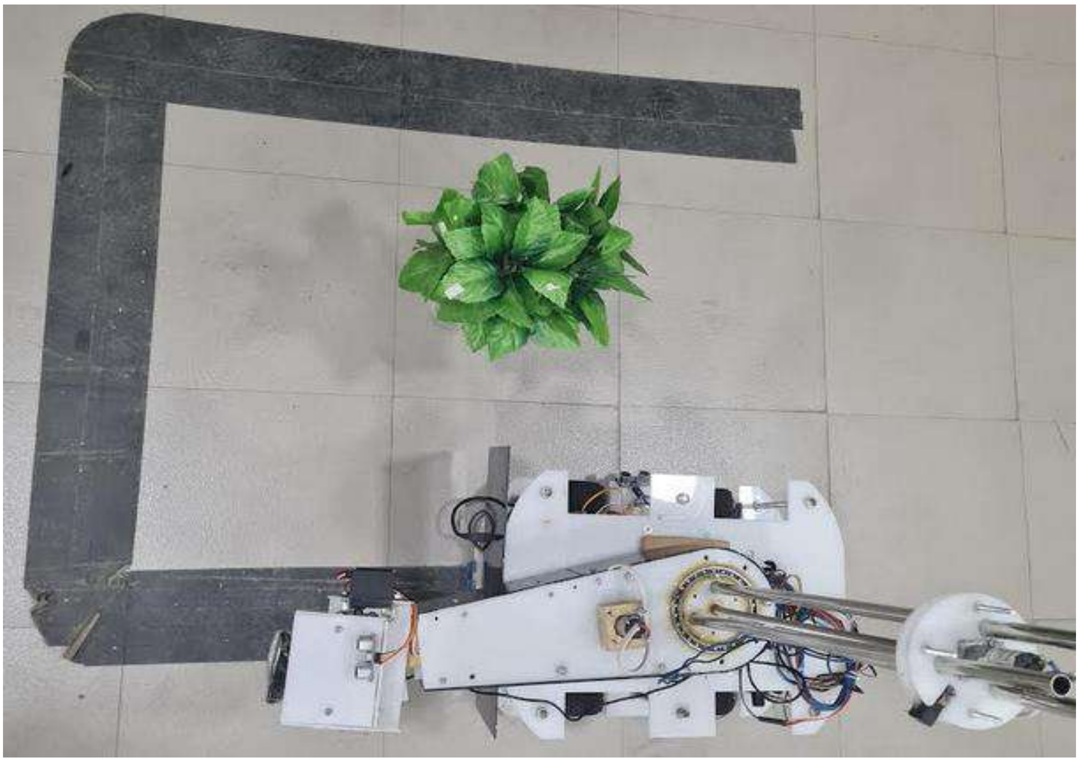


Figure 4: Robotic Arm Movement

2.2.1 SOFTWARE AND HARDWARE DESIGN

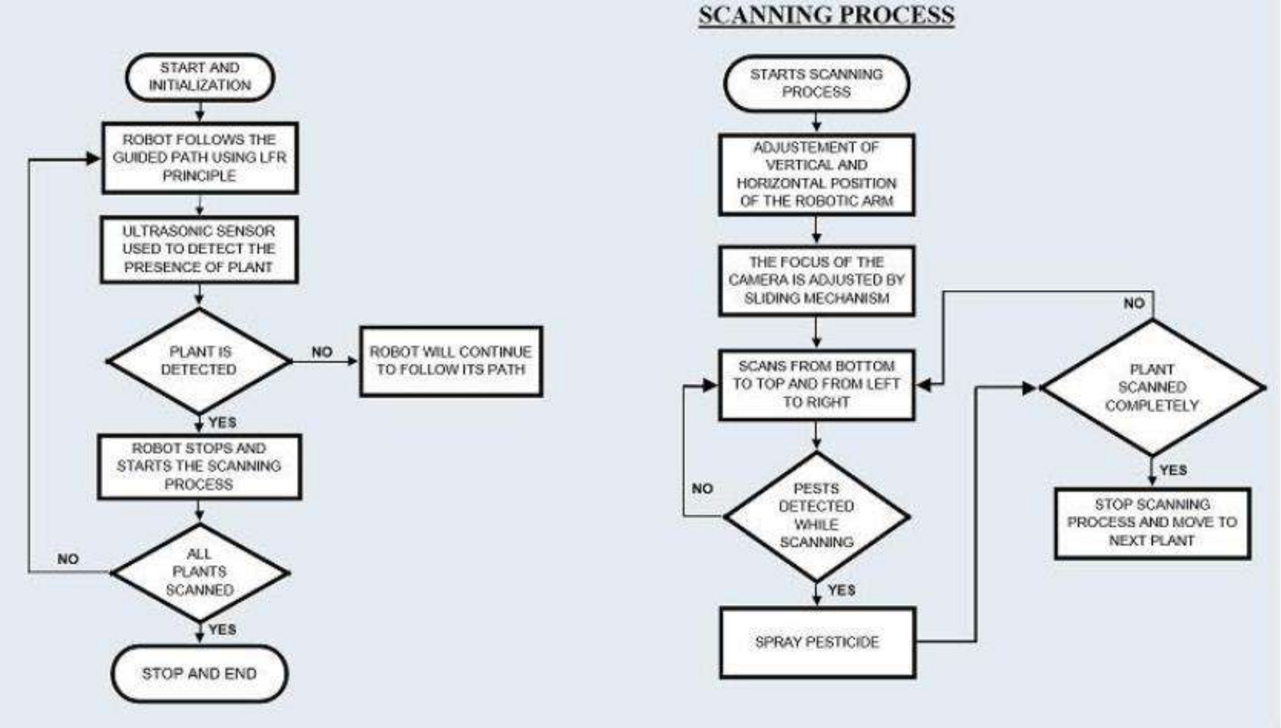
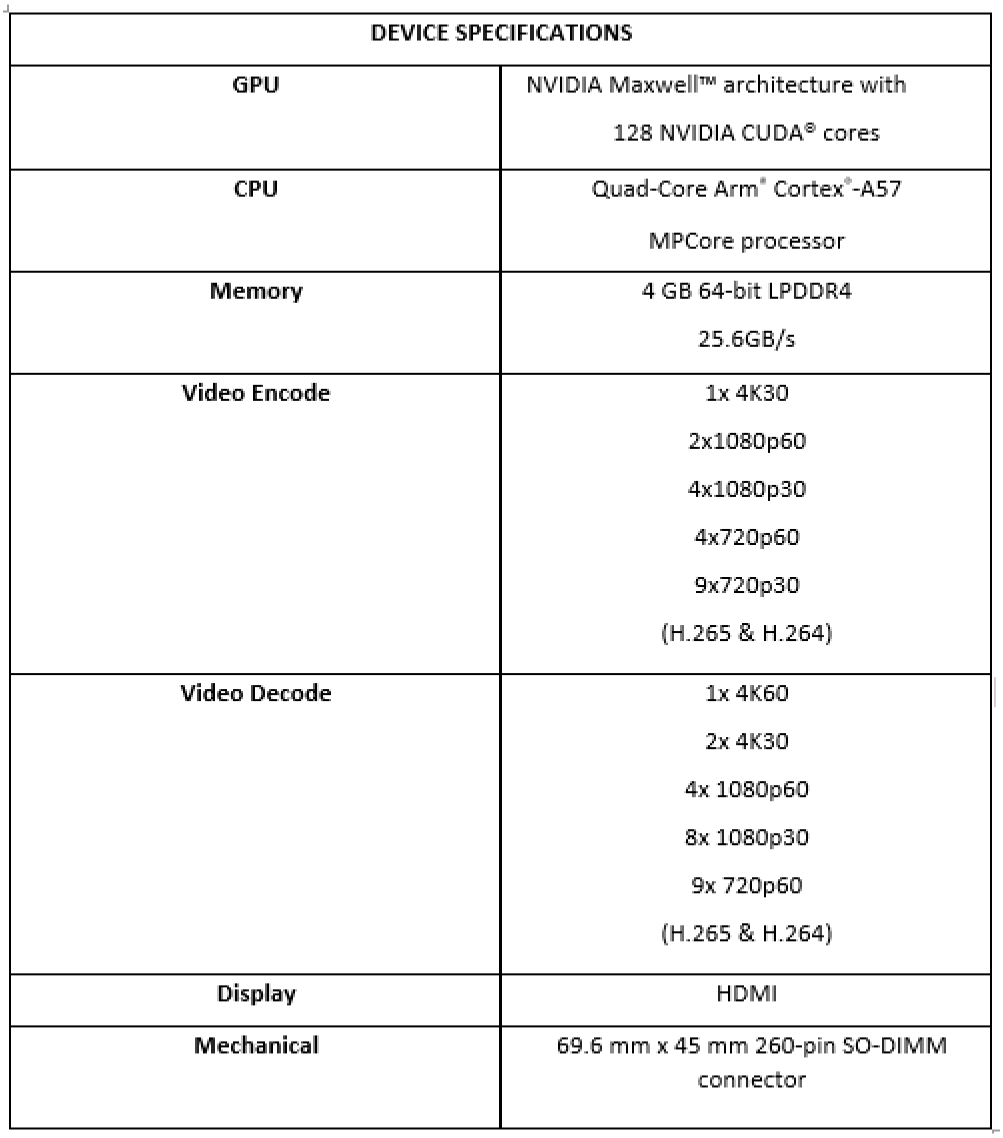

SCANNING PROCESS
START AND INITIALIZATION
ROBOT FOLLOWS THE GUIDED PATH USING LFR PRINCIPLE
ULTRASONIC SENSOR USED TO DETECT THE PRESENCE OF PLANT
PLANT IS DETECTED
NO
ROBOT WILL CONTINUE TO FOLLOW ITS PATH
YES
ROBOT STOPS AND STARTS THE SCANNING PROCESS
ALL PLANTS SCANNED
NO
YES
STOP AND END
STARTS SCANNING PROCESS
ADJUSTEMENT OF VERTICAL AND HORIZONTAL POSITION OF THE ROBOTIC ARM
THE FOCUS OF THE CAMERA IS ADJUSTED BY SLIDING MECHANISM
SCANS FROM BOTTOM TO TOP AND FROM LEFT TO RIGHT
PESTS DETECTED WHILE SCANNING
NO
YES
SPRAY PESTICIDE
PLANT SCANNED COMPLETELY
NO
YES
STOP SCANNING PROCESS AND MOVE TO NEXT PLANT

Figure 5:Design Flow

SOFTWARE DESIGN

Nvidia jetson developer kit is very popular Single Board Computer. It is a mini computer that allows multiple neural networks running in parallel and also it is capable of deploying AI, Machine learning, computer vision and Deep Learning applications. It uses **Linux4Tegra,** based on Ubuntu 18.04 as its operating system. Jetson Nano is used to detect the pests and is the main controller for the robotic arm. Arduino Mega is also used in conjunction with Nano in order to control the dc motors, sense data for vertical , horizontal , rotational movement and for tilt and pan movement of webcam . Arduino software has been installed in Jetson Nano for the continuous monitoring of data received by Arduino and to integrate both controllers together.



DEVICE SPECIFICATIONS	
GPU	NVIDIA Maxwell™ architecture with 128 NVIDIA CUDA® cores
CPU	Quad-Core Arm® Cortex®-A57 MPCore processor
Memory	4 GB 64-bit LPDDR4 25.6GB/s
Video Encode	1x 4K30 2x1080p60 4x1080p30 4x720p60 9x720p30 (H.265 & H.264)
Video Decode	1x 4K60 2x 4K30 4x 1080p60 8x 1080p30 9x 720p60 (H.265 & H.264)
Display	HDMI
Mechanical	69.6 mm x 45 mm 260-pin SO-DIMM connector

Table 1:Device Specifications

HARDWARE DESIGN:

For the Robotic arm structure we have designed the parts using AutoCad software. The design is implemented on Acrylic sheets after laser cutting.

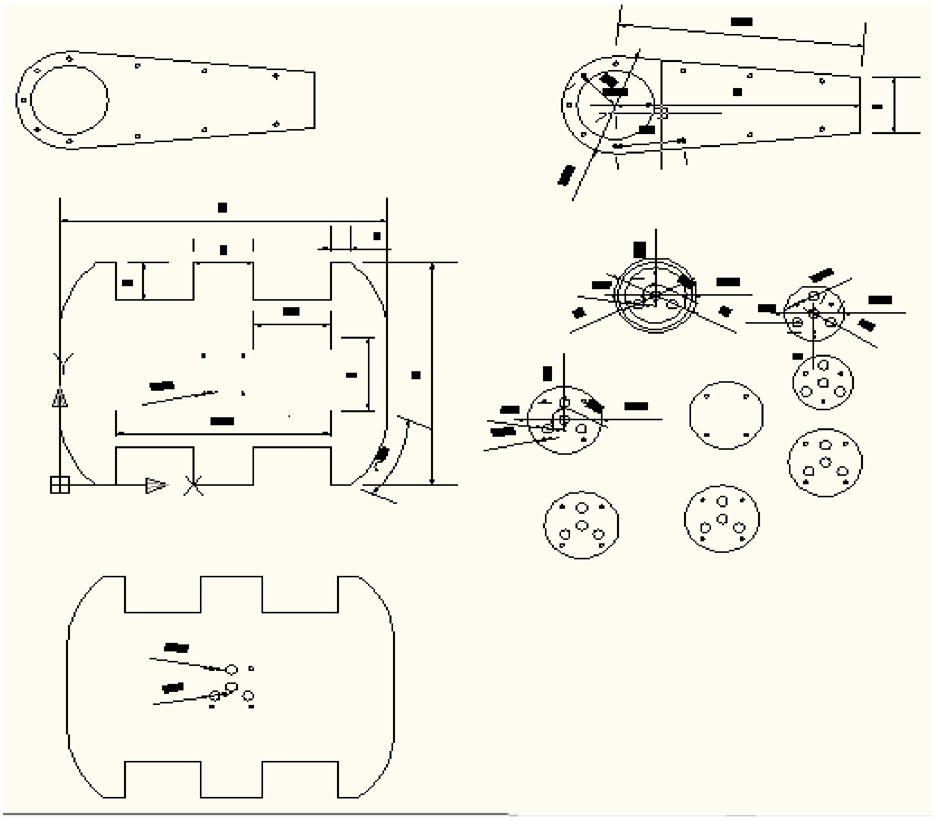


Figure 6:AutoCad design

The robotic arm is designed in such a way that a camera is attached to it along with controllers and motors that will work simultaneously to detect and kill pests. The hardware is designed in a way to move around the field and also move horizontally, vertically and rotationally for proper scanning of the plant.

- **Line Following Robot:**

The robotic arm moves on the phenomena of LFR which uses IR sensors and Arduino microcontroller which is programmed to follow a certain path and stop when an obstacle is detected which will be sensed by ultrasonic sensors.

- **Vertical Movement**

The vertical movement of the robotic arm aims to adjust the height of the arm according to the plant height. DC motors are used along with limit switches to set the limit and move the arm vertically.

- **Rotational Movement**

The gear is connected to the DC motor. The motor rotation results in gear rotation, which will rotate the bearing and hence the arm will rotate. The bearing and gears are attached together by a belt. Rotatory encoders are used which counts the steps the arms takes to rotate. The algorithm is designed to make the arm stop at an angle of 45° (11 steps) and -45° (22 steps) .

- **Sliding Mechanism**

The sliding mechanism of the robotic arm is designed to set the focus of the camera. Using Dc motors the rod will move and will extend the camera attached to it 6" away from the plant as this distance results in a good focus of the camera. Ultrasonic sensor will measure the distance and adjust accordingly.

- **Camera Movement**

Using servo motors to tilt (move vertically) and pan (move horizontally) the camera.

2.2.2 Network Archictectures and Models for transfer Learning

DEEP LEARNING

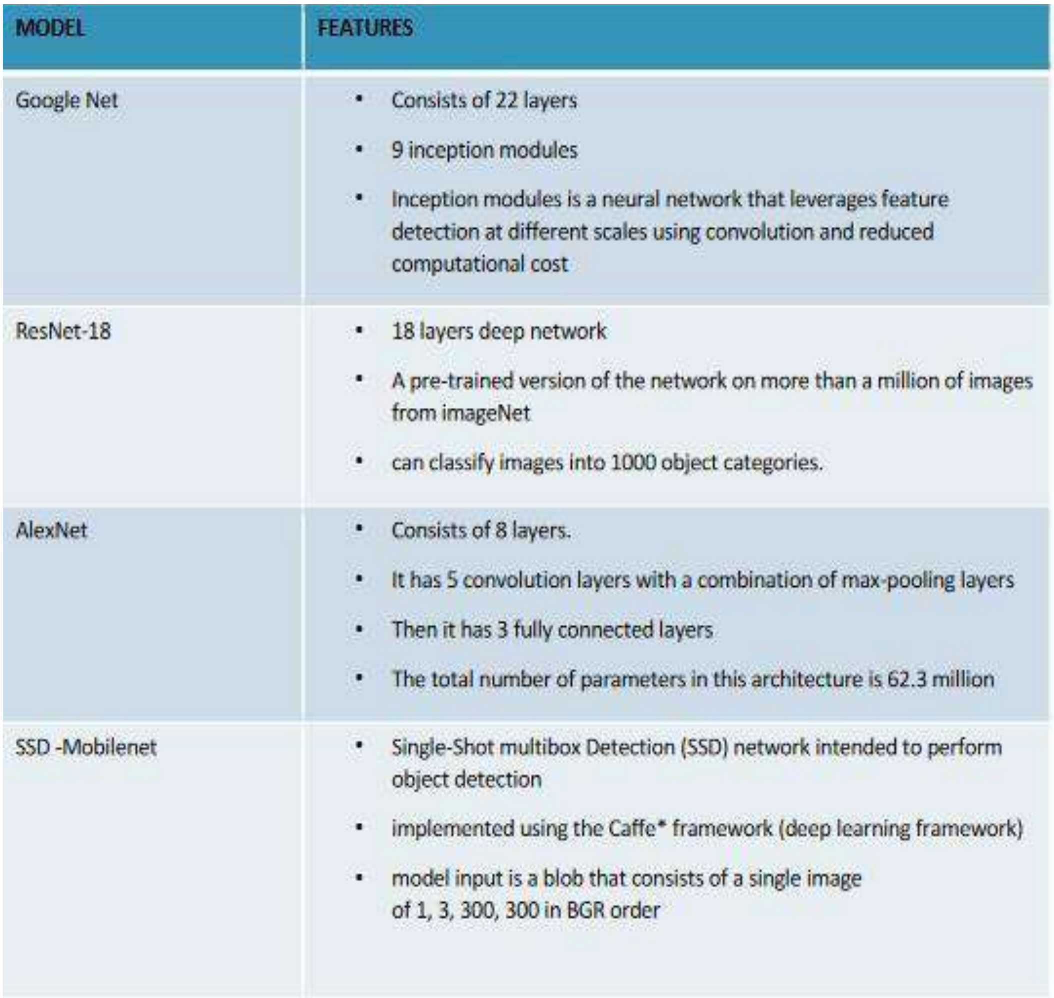
Deep learning has brought advancements in the field of computer vision. Deep learning enables machines to mimic like a human brain . There are several deep learning algorithms such as CNN (Convolution Neural Network) and RNN (Recursive Neural Network).

Each of the algorithms have different features. For example RNN are better implemented in applications that are time dependent where CNN is widely used in image recognition and detection applications. A comparative study showed CNN has shown great results out of all the algorithms and has brough great advancement to the field of computer vision.

CNN – CONVOLUTION NEURAL NETWORK

Convolutional Neural Networks (CNNs) that have brought perfection to this field over the time. They are widely used type of neural networks used in computer vision for recognition and detection of objects .CNN's takes an input image assign importance to some objects and differentiate one form from another. CNN requires the least amount of pre-processing compared to the algorithm. CNN architecture in analogous to the patterns of neurons in human brain.They use convolution phenomena to detect edges from an image. CNN has the following layers:

Convolution Layers , pooling layers and fully connected layers.Within a convolutional layer, the input is first transformed and then passed to the next layer. It uses filter to transform data. A filter is simply a matrix of randomized number values. These layers work together to adapt special features of an image.



MODEL	FEATURES
Google Net	• Consists of 22 layers • 9 inception modules • Inception modules is a neural network that leverages feature detection at different scales using convolution and reduced computational cost
ResNet-18	• 18 layers deep network • A pre-trained version of the network on more than a million of images from imageNet • can classify images into 1000 object categories.
AlexNet	• Consists of 8 layers. • It has 5 convolution layers with a combination of max-pooling layers • Then it has 3 fully connected layers • The total number of parameters in this architecture is 62.3 million
SSD -Mobilenet	• Single-Shot multibox Detection (SSD) network intended to perform object detection • implemented using the Caffe* framework (deep learning framework) • model input is a blob that consists of a single image of 1, 3, 300, 300 in BGR order

Table 2:CNN models

SSD – SINGLE SHOT DETECTOR

To train the dataset we have used SSD MobileNet which is a Convolution Neural Network architecture widely used for object detection. SSD stands for Single Shot Detector which has a single convolution network it uses a base architecture called mobileNet (which are efficient convolution neural networks) and has several convolution layers. SSD takes only one shot in order to detect several objects in an image while some others need more than 1 shot which makes it much faster. It has a feed-forward convolution network , which keeps producing boxes and scores around desired objects. It extracts feature map and the apply convolution in order to detect any objects present.

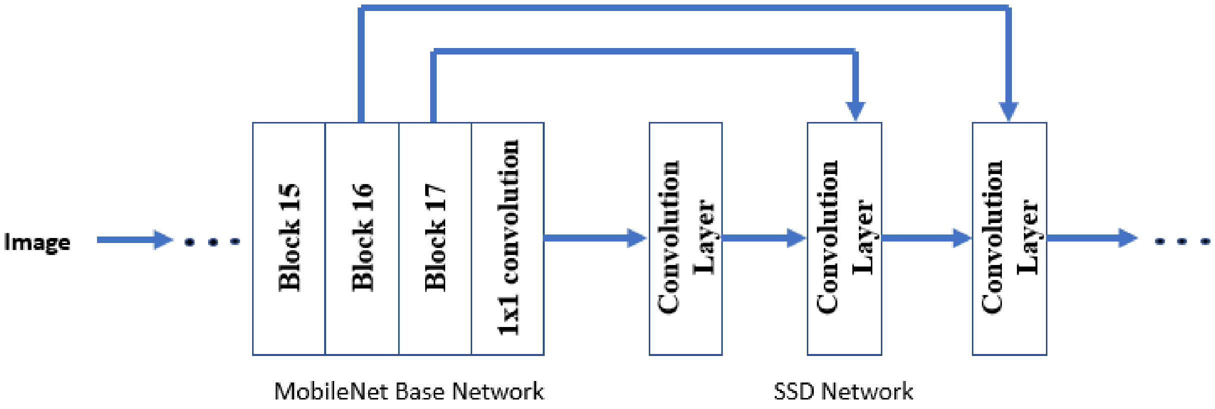


Figure 7: MobileNet SSD architecture

CHAPTER 3

DESIGN IMPLMENTATION

3.1 Hardware implementation

ROBOT WORKING

Our aim was to design a robotic arm for the pest detection and pest killing purpose. To achieve that aim we've designed a movable robot in a way that it will move around a plant to cover the whole plant for scanning from bottom to top. For this scanning purpose we've fitted a Logitech c270 webcam through a sliding mechanism. This sliding mechanism is used to adjust the camera focus for pests that would be detect by a sensor. For vertical movement of arm a sliding rod has been used which is attached to the DC motors. As the motor rotates the rod will rotate and the arm will move vertically. The arm will stop at desired instances as set in the algorithm. Limits switches are used to set limit for the vertical movement. At each instance the arms stops vertically , the rotational movement is put into action. The gear is connected to the motor. The motor rotation results in gear rotation, which will rotate the bearing and hence the arm will rotate. The The bearing and gears are attached together by a belt. The main component used for this part are rotatory encoders which counts the steps the arms takes to rotate. It is set to stop after certain steps and as it stops horizontal movement of the arm is put into action. It uses a sliding mechanism which is mainly to set the focus of the camera attached to it. The camera's tilt and Pan movement is controlled using servo motors.

When it stops the camera starts detecting and sprays pesticide if pests are detected if not it continues to move to next step.

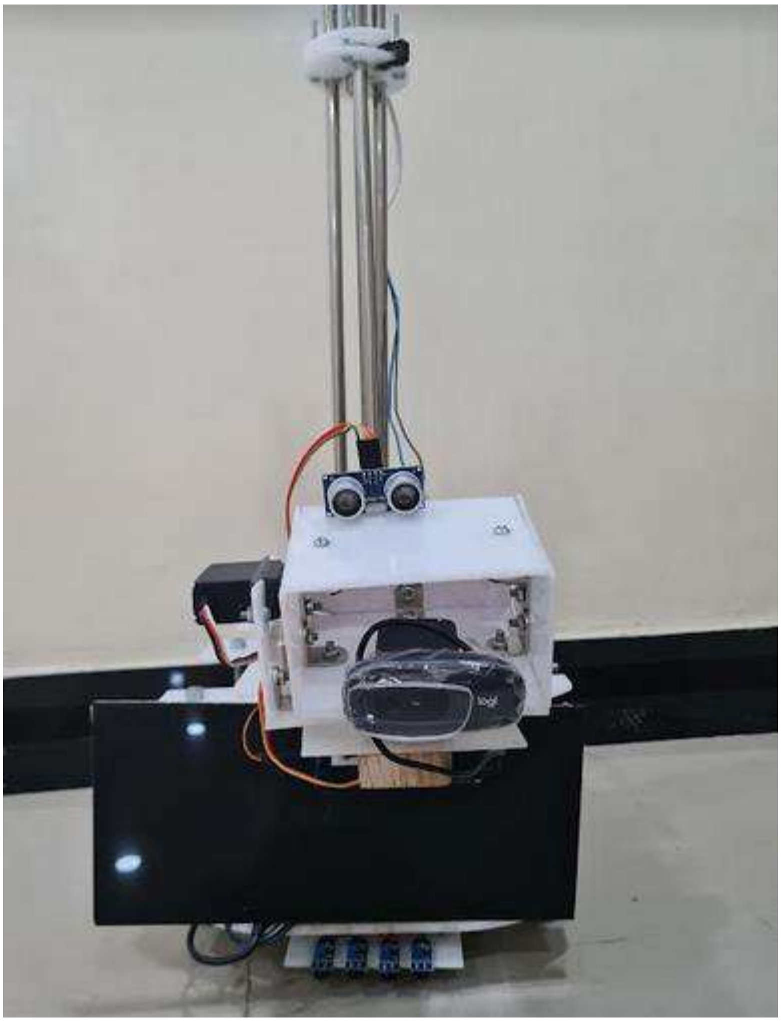


Figure 8:Robotic Arm

3.2 Training the Jetson Nano

The Aim of the project is to detect the presence of pest on the leaves. In order to achieve it we first need to train our device for the pests. To train the device for the pest we will collect a dataset which will contain set of pictures of each kind of pest that we will manually take using Logitech c270 webcam

Using the method of transfer learning the device will be trained for the dataset.

Procedure of collecting the dataset will be discussed in the upcoming sections.

3.2.1 Collecting the Dataset

We have trained for the following artificial pests:

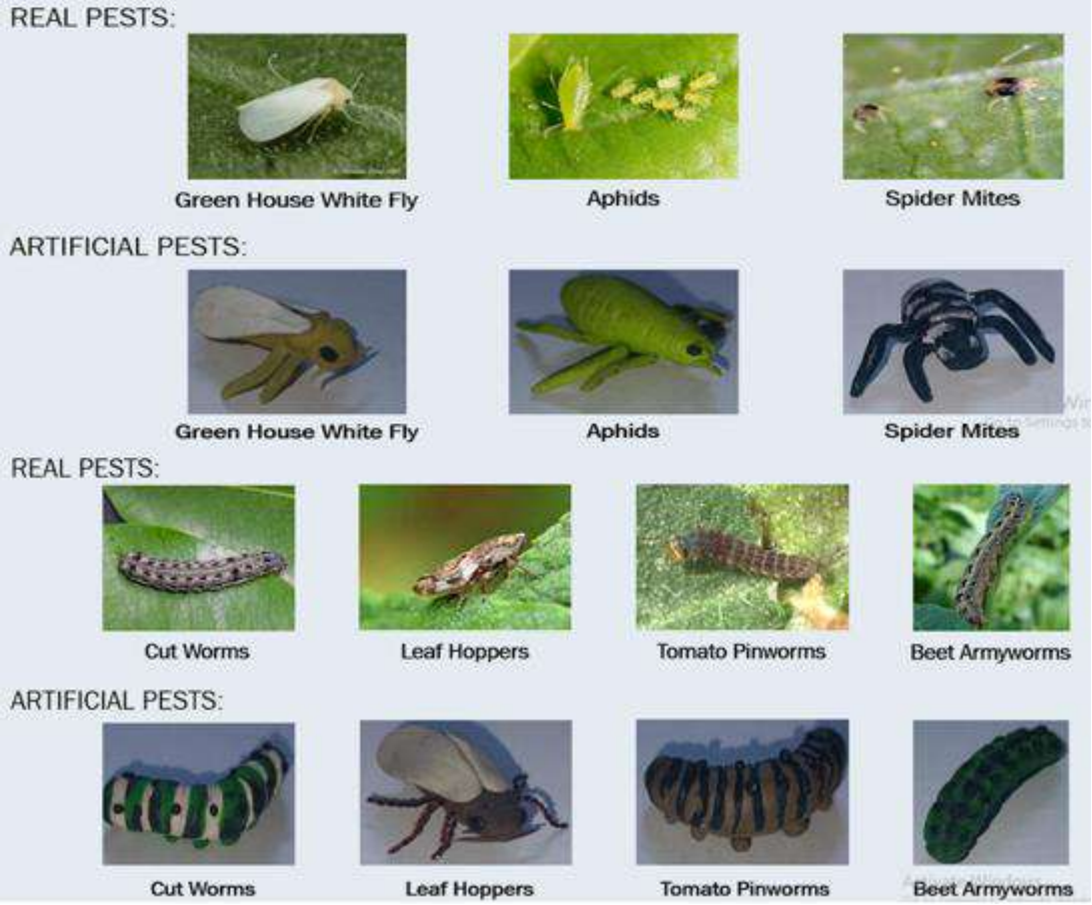


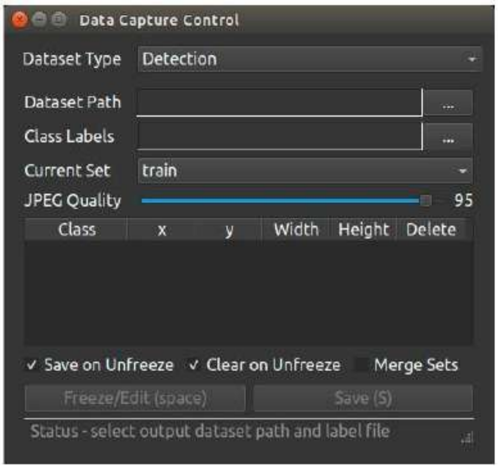
Figure 9:Dataset

LIST OF PESTS
Greenhouse Whitefly
Tomato Pinworm
Cutworms
Spider Mites
Aphids
Leaf Hoppers
Beet Armyworm

Table 3:Pest Labels

3.2.2 Creating Dataset

A GUI named “Data Capture Control” appeared for setting the instruction to the device that for what purpose we are willing to take pictures. For example: for image classification we can choose **classification** option in **data type** but since our aim is to detect the pests so we selected **detection** in **data type**. Next we defined dataset path and class labels.



Data Capture Control

Dataset Type: Detection

Dataset Path

Class Labels

Current Set: train

JPEG Quality: 95

Class	x	y	Width	Height	Delete

✓ Save on Unfreeze ✓ Clear on Unfreeze □ Merge Sets

Status - select output dataset path and label file

Figure 10:Data Capture tool

- **Save on Unfreeze** - automatically save the data when Freeze/Edit is unfreezed
- **Clear on Unfreeze** - automatically remove the previous bounding boxes on unfreeze
- **Merge Sets** - save the same data across the train, val, and test sets
- **Current Set** - select from train/val/test sets ○ for object detection, you need at least train and test sets ○ although if you check Merge Sets, the data will be replicated as train, val, and test
- **JPEG Quality** - control the encoding quality and disk size of the saved images

For dataset path and class labels we created a new directory in the following path:

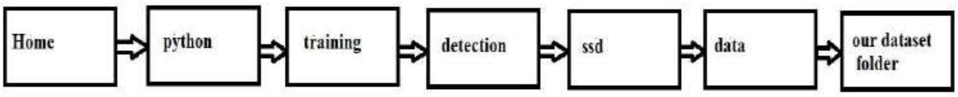


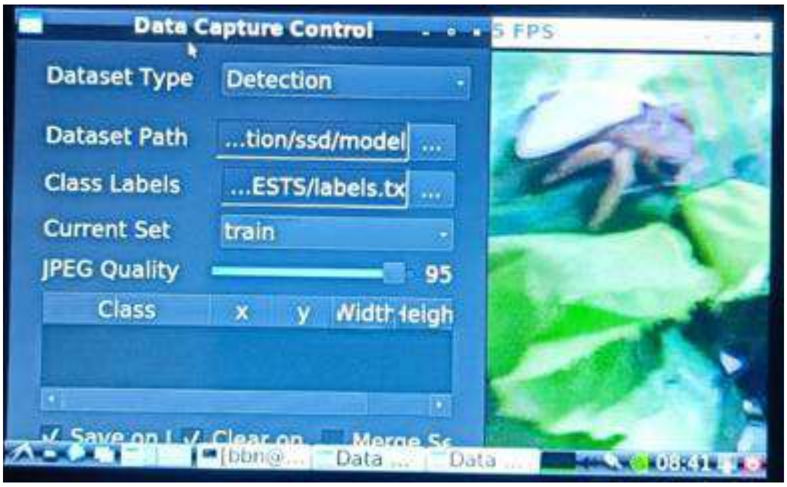
Figure 11:Dataset Path

Creating Label Files

In our dataset folder we created label.txt file and opened that label file in text editor and typed the name of the classes (pests) that we were going to detect.

Process of taking pictures

After setting up all these instruction we made dataset by taking pictures. To take pictures we clicked on the freeze icon to freeze camera frame and then we drew bounding boxes around pests then unfroze the image and saved it and took multiple pictures of differ ent viewpoints and orientations of the pests. The dataset was made by repeating the same procedure. Bounding box were made as close as possible to the pests.. We also check-marked the merge set option that duplicates the data between train, valid and tests datasets. (which saves duplicacy of dataset of the captured data across train, val and test sets.)



Data Capture Control

Dataset Type: Detection

Dataset Path: ...tion/ssd/model

Class Labels: ...ESTS/labels.tx

Current Set: train

JPEG Quality: 95

Class	x	y	Width	Heigh

☑ Save on I ☑ Clear on □ Merge Se

Figure 12:Creating Dataset

3.2.3 Training and testing methods

Some packages and libraries for training:

PyTorch:

PyTorch is a machine learning framework. It is a Python package that provides two high-level features:

- Tensor computation (like NumPy) with strong GPU acceleration
- Deep neural networks built on a tape-based autograd system

TENSOR FLOW

It is an open-source library used for machine learning and AI. It can be used for several tasks but mostly used for training and inference of deep neural networks.

ONNX:

The Open Neural Network Exchange is an open-source artificial intelligence ecosystem of technology companies and research organizations that create standards for representing machine learning algorithms and software tools to enhance the growth of AI sector..

There are two methods to train the jetson nano that will be discussed in the upcoming sections.

- **Running the Docker Container**

To train the model through docker container we used ssd.py script that uses pytorch at its backend. We first trainer for 30 epoch cycle using the Docker container.

Converting model from pytorch to onnx

After training was completed our model from pytorch to ONNX using this code 3.

Testing

After converting the trained model from pytorch to ONNX, we tested our dataset (that we had already made) on a live camera stream. It was detecting the pest with 50% to 60% accuracy.

- **Building the Project from source**

This is another training method in which we just installed base model which provide some packages that support object detection process and rest of the training method remains the same as in the docker container. We now trained the jetson nano by this method for 100 epoch cycles .

Converting model from pytorch to onnx

After training we exported our model from pytorch to ONNX

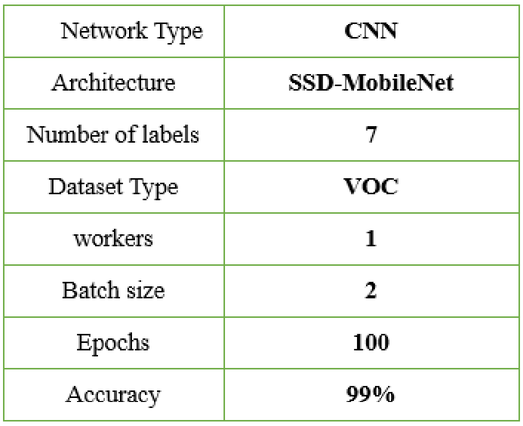
Testing

Secondly we trained the device by "**Building the project from source**" method and then tested our dataset on a live camera stream so it was detecting the pests with about 98% accuracy.

CHAPTER 4

RESULTS AND DISCUSSIONS

4.1 Results



Network Type	**CNN**
Architecture	**SSD-MobileNet**
Number of labels	**7**
Dataset Type	**VOC**
workers	**1**
Batch size	**2**
Epochs	**100**
Accuracy	**99%**

Table 4: Training Parameters

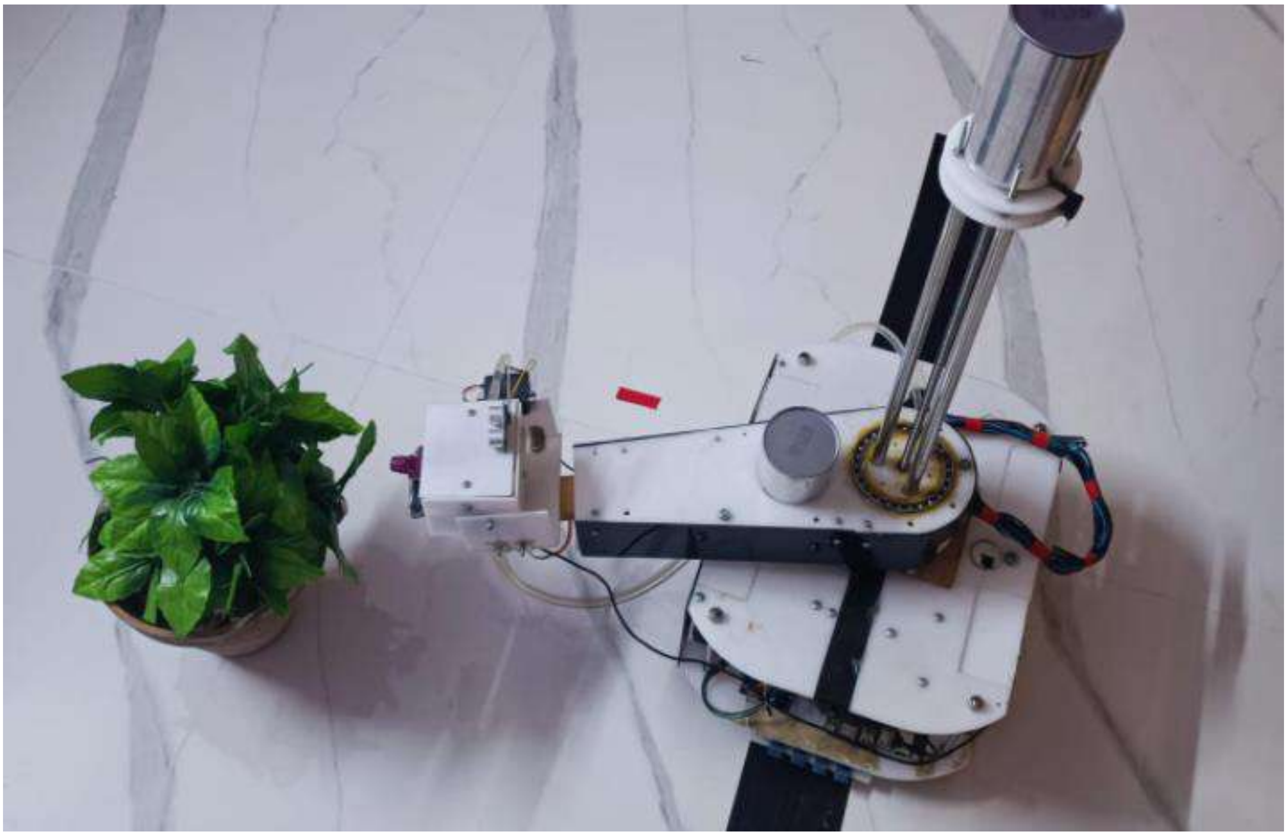


Figure 13:Pest Detected by the Robotic Arm

Our dataset consisted of 1000 pictures and it was trained for 100 epochs. After training the results were observed. Bounding boxes were drawn over detected objects which also mentioned the name of the pests. It shows that the Greenhouse whitefly was detected at an accuracy of 98% , tomato pinworm at an accuracy 68.5% and cutworms with an accuracy of 99.9 % as illustrated in the image below.

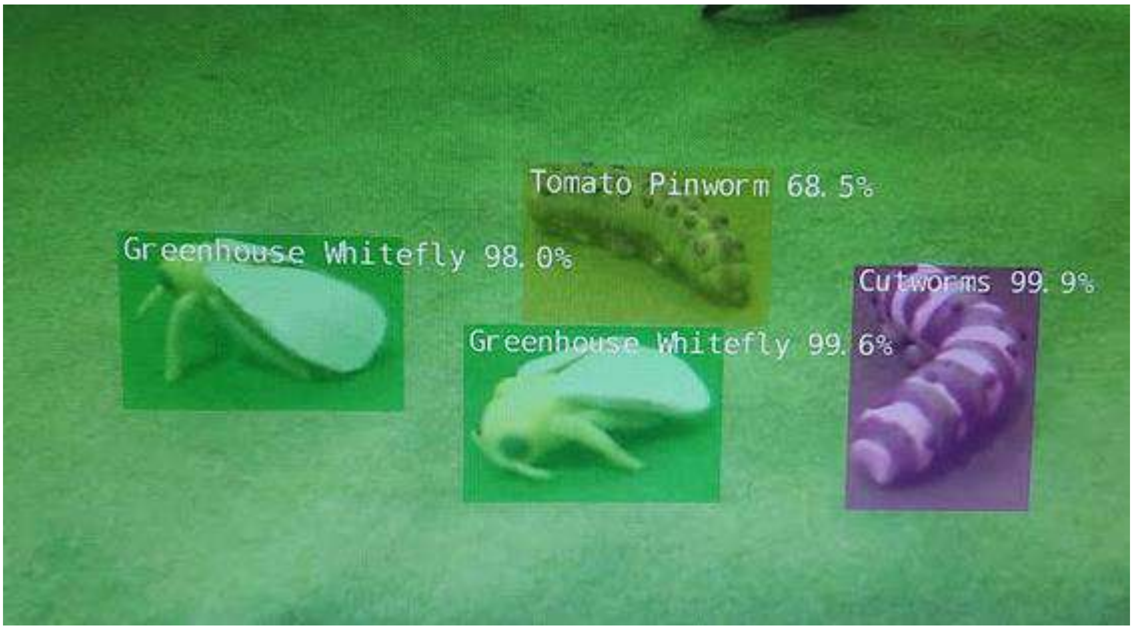


Figure 14:Pest Detection

The network was trained for 30 epochs and 100 epochs on the same dataset and accuracy was observed. Training the dataset for 30 epochs resulted in an accuracy 63% and for 100 epochs the accuracy was 99%. The results are illustrated in the bar chart below.

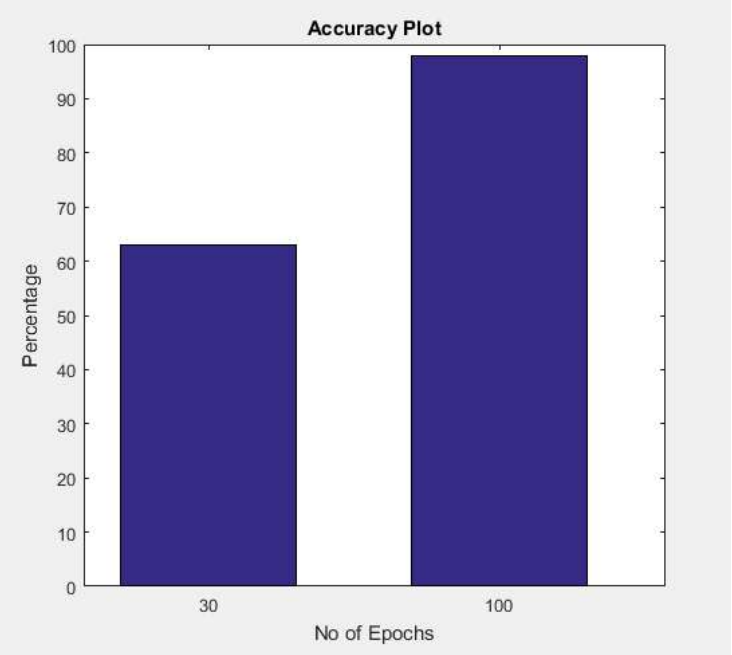


Figure 15:Percentage Accuracy Bar Chart

The accuracy plot for Leap Hoppers shows that the best accuracy was achieved at a distance of 11cm.

As illustrated in the graph below.

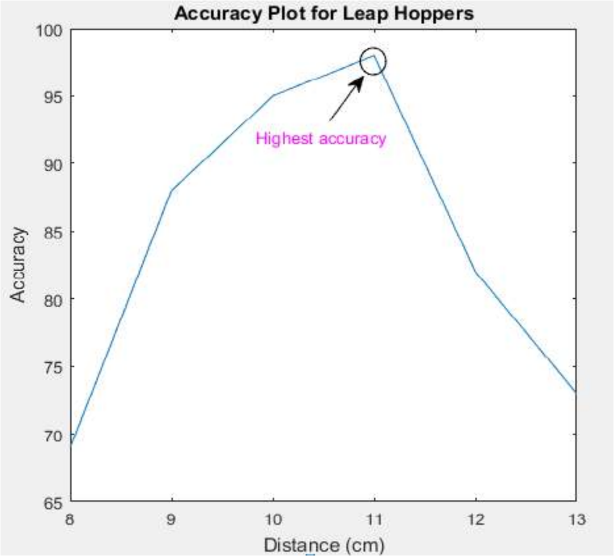


Figure 16:Accuracy Plot for Leaf Hoppers

The accuracy plot for Aphids shows that the best accuracy was achieved at a distance of 11cm.

As illustrated in the graph below.

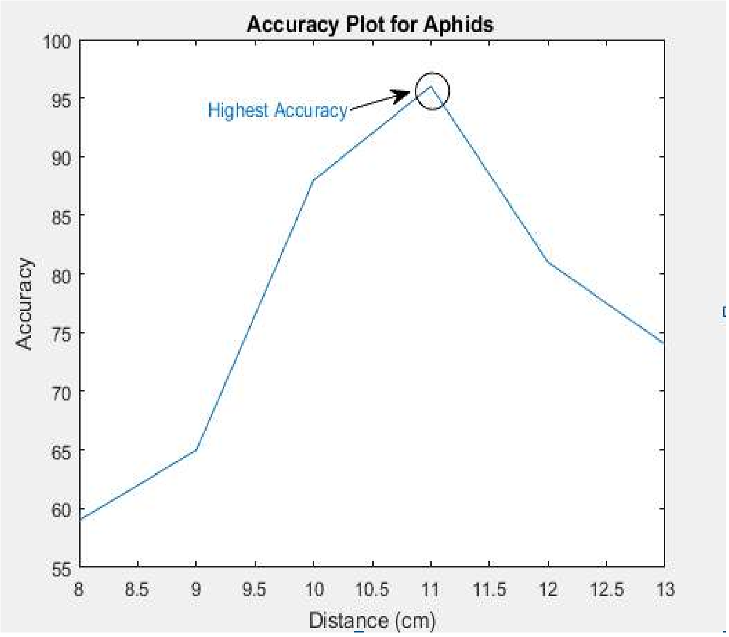


Figure 17:Accuracy Plot for Aphids

The accuracy plot for GreenHouse Whitefly shows that the best accuracy was achieved at a distance between 12cm and 13cm.

As illustrated in the graph below.

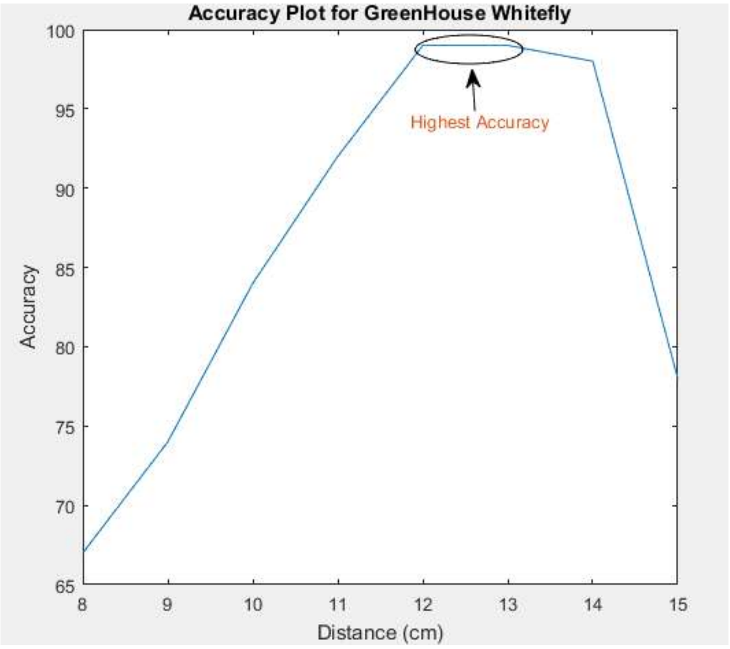


Figure 18:Accuracy Plot for GreenHouse Whitefly

The accuracy plot for Spider Mites shows that the best accuracy was achieved at a distance of 15cm.

As illustrated in the graph below.

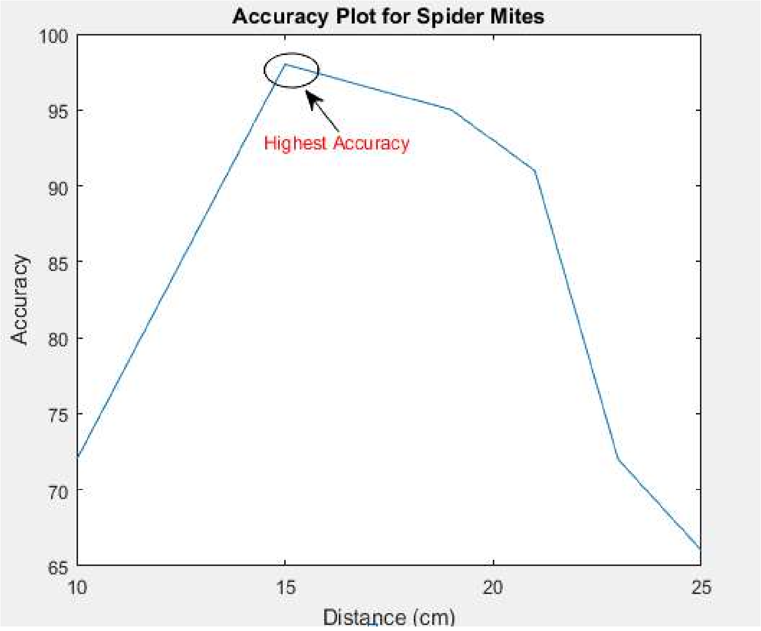


Figure 19:Accuracy Plot for Spider Mites

The accuracy plot for Beet Armyworms shows that the best accuracy was achieved at a distance 12cm and 13cm.

As illustrated in the graph below.

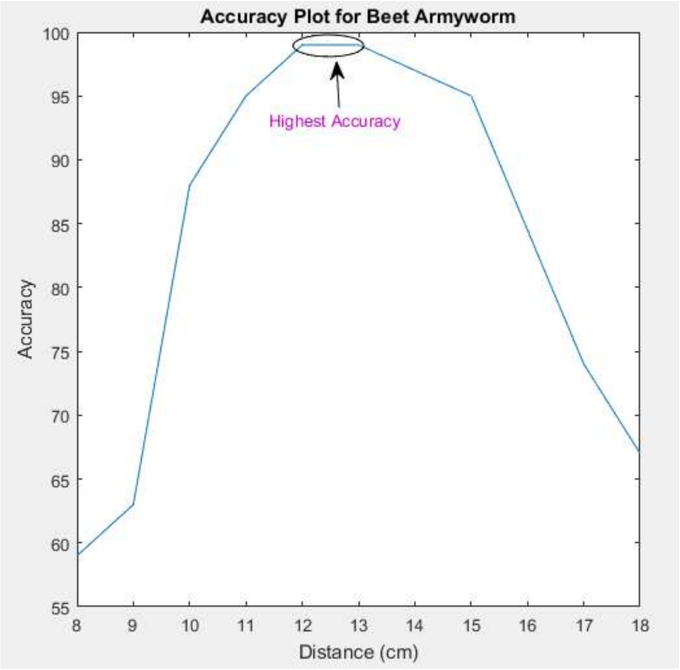


Figure 20:Accuracy Plot for Beet Armyworm

4.2 Discussion

Using the concepts of machine learning the device was trained for our created dataset consisting of 1000 pictures. The dataset was first trained for 30 epochs(training cycle) which resulted in an accuracy of 63%. To improve the accuracy, The dataset was then trained for 100 epochs which resulted in an accuracy of 99%. This is the best accuracy that can be achieved.

A few observations were made on how the distance of the camera affects the accuracy of detection Each pest was observed for how the distance affects the detection accuracy for that particular pest. The accuracy of Leap Hoppers and Aphids is best when the distance is 11cm . The accuracy of Green House Whitefly and Beet Armyworms best at a distance between 12cm and 13cm. The accuracy of Spider Mites is best at a distance of 15cm.

Distance is one parameter that affects the accuracy whereas training cycles is another parameter for a comparative study regarding accuracy. There may be several parameters that can be varied and its affect can be observed but for our work we have analyzed our results based on distance and number of training cycles.

CHAPTER 5
CONCLUSIONS AND RECOMMENDATIONS

5.1 Conclusion

Agricultural sector is the largest contributor to Pakistan's economy. There are several environmental factors that can inhibit or destroy the growth of plants and compromise crop quality such as pests. Poor quality can lead to low demand and largely affect the economy. To counter this problem, Computer vision and Machine learning can be used to detect multiple pests present and then to spray pesticide on those specific parts of the plants on which the pests are present. The Robotic Arm is based on LFR principle to move around fields and scan all plants. Applying the concepts of AI and machine learning, the Jetson Nano has been trained to use pretrained networks through transfer learning to recognize.The dataset has been collected using the camera capture tool and nano has been trained on the dataset. Using SSD-MobileNet as the CNN architecture which gives best results and training on 100 epochs. 99% accuracy was achieved in pest detection.

REFERENCES

[1] G. L. Tenório et al., "Comparative Study of Computer Vision Models for Insect Pest Identification in Complex Backgrounds," 2019 12th International Conference on Developments in eSystems Engineering (DeSE), 2019, pp. 551-556, doi: 10.1109/DeSE.2019.00106.

[2] Patel, Deven J., and Nirav Bhatt. "Insect identification among deep learning's meta-architectures using TensorFlow." *Int. J. Eng. Adv. Technol* 9.1 (2019): 1910-1914.

[3] Mahesh, Batta. "Machine learning algorithms-a review." *International Journal of Science and Research (IJSR).[Internet]* 9 (2020): 381-386.

[4] A. Suárez, R. S. Molina, G. Ramponi, R. Petrino, L. Bollati and D. Sequeiros, "Pest detection and classification to reduce pesticide use in fruit crops based on deep neural networks and image processing," 2021 XIX Workshop on Information Processing and Control (RPIC), 2021, pp. 1-6, doi: 10.1109/RPIC53795.2021.9648485.

[5] B. Jayanthi, T. A. Priyanka, V. B. Shalini and R. K. Grace, "Pest Detection in Crops Using Deep Neural Networks," 2022 8th International Conference on Advanced Computing and Communication Systems (ICACCS), 2022, pp. 29-32, doi: 10.1109/ICACCS54159.2022.9785155.

[6] Liu, Jun, and Xuewei Wang. "Plant diseases and pests detection based on deep learning: a review." *Plant Methods* 17.1 (2021): 1-18.

[7] J. Tang, X. Peng, X. Chen and B. Luo, "An Improved Mobilenet-SSD Approach For Face Detection," 2021 40th Chinese Control Conference (CCC), 2021, pp. 8072-8076, doi: 10.23919/CCC52363.2021.9549245.

[8] H. Hong, J. Lin and F. Huang, "Tomato Disease Detection and Classification by Deep Learning," 2020 International Conference on Big Data, Artificial

Intelligence and Internet of Things Engineering (ICBAIE), 2020, pp. 25-29, doi: 10.1109/ICBAIE49996.2020.00012.

[9] Voulodimos, Athanasios, et al. "Deep learning for computer vision: A brief review." *Computational intelligence and neuroscience* 2018 (2018).

[10] Redmon, Joseph, et al. "You only look once: Unified, real-time object detection." *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016.

[11] Zou, Zhengxia, et al. "Object detection in 20 years: A survey." *arXiv preprint arXiv:1905.05055* (2019).

[12] Wang, Sun-Chong. "Artificial neural network." *Interdisciplinary computing in java programming*. Springer, Boston, MA, 2003. 81-100.

[13] Badamasi, Yusuf Abdullahi. "The working principle of an Arduino." *2014 11th international conference on electronics, computer and computation (ICECCO)*. IEEE, 2014.

[14] Cass, Stephen. "Nvidia makes it easy to embed AI: The Jetson nano packs a lot of machine-learning power into DIY projects-[Hands on]." *IEEE Spectrum* 57.7 (2020): 14-16.

[15] T. Kasinathan, et al., "Insect classification and detection in field crops using modern machine learning techniques." *Information Processing in Agriculture* 8.3 (2021): 446-457.

[16] Rehman, Saif Ur, Muhammad Rashid Razzaq, and Muhammad Hadi Hussian. "Training of ssd (single shot detector) for facial detection using nvidia jetson nano." *arXiv preprint arXiv:2105.13906* (2021).

[17] Boissard, Paul, Vincent Martin, and Sabine Moisan. "A cognitive vision approach to early pest detection in greenhouse crops." *computers and electronics in agriculture* 62.2 (2008): 81-93.

[18] Selvaraj, Michael Gomez, et al. "AI-powered banana diseases and pest detection." *Plant Methods* 15.1 (2019): 1-11.

[19] Türkoğlu, Muammer, and Davut Hanbay. "Plant disease and pest detection using deep learning-based features." *Turkish Journal of Electrical Engineering and Computer Sciences* 27.3 (2019): 1636-1651.

[20] Nam, Nguyen Tuan, and Phan Duy Hung. "Pest detection on traps using deep convolutional neural networks." *Proceedings of the 2018 International Conference on Control and Computer Vision*. 2018.

[21] Nagar, Harshita, and R. S. Sharma. "A comprehensive survey on pest detection techniques using image processing." *2020 4th International Conference on Intelligent Computing and Control Systems (ICICCS)*. IEEE, 2020.

[22] D. Arora, M. Garg and M. Gupta, "Diving deep in Deep Convolutional Neural Network," 2020 2nd International Conference on Advances in Computing, Communication Control and Networking (ICACCCN), 2020, pp. 749-751, doi:10.1109/ICACCCN51052.2020.9362907.

[23] Prathibha, G. P., et al. "Early pest detection in tomato plantation using image processing." *International Journal of Computer Applications* 96.12 (2014).

[24] O'Shea, Keiron, and Ryan Nash. "An introduction to convolutional neural networks." *arXiv preprint arXiv:1511.08458* (2015).

[25] Suganuma, Masanori, Shinichi Shirakawa, and Tomoharu Nagao. "A genetic programming approach to designing convolutional neural network architectures." *Proceedings of the genetic and evolutionary computation conference*. 2017.

[26] Womg, Alexander, et al. "Tiny SSD: A tiny single-shot detection deep convolutional neural network for real-time embedded object detection." *2018 15th Conference on Computer and Robot Vision (CRV)*. IEEE, 2018.

[27] Dhillon, Anamika, and Gyanendra K. Verma. "Convolutional neural network: a review of models, methodologies and applications to object detection." *Progress in Artificial Intelligence* 9.2 (2020): 85-112.

[28] Younis, Ayesha, et al. "Real-time object detection using pre-trained deep learning models MobileNet-SSD." *Proceedings of 2020 the 6th international conference on computing and data engineering*. 2020.

[29] Ghoury, Shekofa, Cemil Sungur, and Akif Durdu. "Real-time diseases detection of grape and grape leaves using faster r-cnn and ssd mobilenet architectures." *International conference on advanced technologies, computer engineering and science (ICATCES 2019)*. 2019.

[30] Chiu, Yu-Chen, et al. "Mobilenet-SSDv2: An improved object detection model for embedded systems." *2020 International conference on system science and engineering (ICSSE)*. IEEE, 2020.

APPENDIX A

The code through which the device was trained through docker contatiner for 100 epoch cycles.

```
root@bbn-desktop:/jetson-inference/python/training/detection/ssd#          python3
train_ssd.py              --dataset-type=voc              --data=data/NewPest              --
modeldir=models/NewPestTestOne --batch-size=2 --workers=1 --epochs=100
```

The model trained via pytorch was then converted to onnx.

```
root@bbn-desktop:/jetsoninference/python/training/detection/ssd#python3
        onnx_export.py          -model-dir=models/NewPestTestOne
```

With the below code we tested our model by doing pests detection.

```
root@bbn-desktop:/jetson-inference/python/training/detection/ssd#   detectnet   --
model=models/NewPestTestOne/ssd-mobilenet.onnx-
labels=models/NewPestTestOne/labels.txt              --input-blob=input_0              --
outputcvg=scores --output-bbox=boxes /dev/video0
```

The code used to train the device by building the project from sorce for 100 epoch cycles.

```
bbn@bbn-desktop:/jetsoninference/python/training/detection/ssd$           python3
train_ssd.py           --datasettype=voc           --data=data/NewPest           --model-
dir=models/NewPestTestOne -batch-size=2 --workers=1 --epochs=100
```

Pytorch trained model was converted to onnx

```
bbn@bbn-desktop:/jetsoninference/python/training/detection/ssd$           python3
onnx_export.py -model-dir=models/NewPestTestOne
```

The results were achieved in testing session using the below code.

```
bbn@bbn-desktop:/jetson-
inference/python/training/detection/ssd$              detectnet              -
model=models/NewPestTestOne/ssd-mobilenet.onnx                               -
labels=models/NewPestTestOne/labels.txt         --input-blob=input_0         --
outputcvg=scores --output-bbox=boxes /dev/video0
```

APPENDIX B

Python code

```
import jetson.inference
import jetson.utils
import time
import cv2
import numpy as np
import serial

timeStamp=time.time()
fpsFilt=0
net=jetson.inference.detectNet('ssd-mobilenet-v2',['--model=/home/bbn/jetson-inference/python/training/detection/ssd/models/NewPestTestOne/ssd-mobilenet.onnx','--labels=/home/bbn/jetson-inference/python/training/detection/ssd/models/NewPestTestOne/labels.txt','--input-blob=input_0','--output-cvg=scores','--output-bbox=boxes'],threshold=0.2)
dispW=1280
dispH=720
flip=2
font=cv2.FONT_HERSHEY_SIMPLEX
item=""
totalpestneutralized=0

# Gstreamer code for improvded Raspberry Pi Camera Quality
#camSet='nvarguscamerasrc wbmode=3 tnr-mode=2 tnr-strength=1 ee-mode=2 ee-strength=1 ! video/x-raw(memory:NVMM), width=3264, height=2464, format=NV12, framerate=21/1 ! nvvidconv flip-method='+str(flip)+' ! video/x-raw, width='+str(dispW)+', height='+str(dispH)+', format=BGRx ! videoconvert ! video/x-raw, format=BGR ! videobalance contrast=1.5 brightness=-.2 saturation=1.2 ! appsink drop=true'
#camSet='nvarguscamerasrc ! video/x-raw(memory:NVMM), width=3264, height=1848, format=NV12, framerate=28/1 ! nvvidconv flip-method='+str(flip)+'
```

```
! video/x-raw, width='+str(dispW)+', height='+str(dispH)+', format=BGRx !
videoconvert ! video/x-raw, format=BGR ! appsink'
#cam=cv2.VideoCapture(camSet)
#cam=jetson.utils.gstCamera(dispW,dispH,'0')

cam=cv2.VideoCapture('/dev/video0')
#cam.set(cv2.CAP_PROP_FRAME_WIDTH, dispW)
#cam.set(cv2.CAP_PROP_FRAME_HEIGHT, dispH)

#cam=jetson.utils.gstCamera(dispW,dispH,'/dev/video1')
#display=jetson.utils.glDisplay()
#while display.IsOpen():
with serial.Serial('/dev/ttyACM0', 9600, timeout=10) as ser:
    while True:

        #img, width, height= cam.CaptureRGBA()
        ret,img = cam.read()
        #height=img.shape[0]
        # #width=img.shape[1]
        # #print (height)
        # #print (width)
        frame=cv2.cvtColor(img,cv2.COLOR_BGR2RGBA).astype(np.float32)
        frame=jetson.utils.cudaFromNumpy(frame)
        detections=net.Detect(frame, 1280, 720)
        for detect in detections:
            print(detect)
            ID=detect.ClassID
            top=int(detect.Top)
            left=int(detect.Left)
            bottom=int(detect.Bottom)
            right=int(detect.Right)
            item=net.GetClassDesc(ID)
            print(item,top,left,bottom,right)
            cv2.rectangle(img,(left,top),(right,bottom),(0,255,0),1)
            ObjectWidth=right-left
            ObjectWidth=ObjectWidth/2
            centerPoint=ObjectWidth+left
            print(centerPoint)
            if centerPoint>=300 and centerPoint<=315:
                print('SPRAY')
                ser.write(bytes('SPRAY','utf-8'))
                time.sleep(2.5)
                totalpestneutralized=totalpestneutralized+1



        #display.RenderOnce(img,width,height)

        dt=time.time()-timeStamp
        timeStamp=time.time()
        fps=1/dt
        fpsFilt=.9*fpsFilt + .1*fps
        #print(str(round(fps,1))+' fps')
        cv2.putText(img,str(round(fpsFilt,1))+'fps ',(0,30),font,1,(0,0,255),2)
        cv2.putText(img,item,(0,60),font,1,(0,0,255),2)
        cv2.putText(img,'Total Pest killed'+totalpestneutralized,(0,80),font,1,(0,0,255),2)

        frame1=cv2.line(img, pt1=(320, 0), pt2=(320, 480), color=(0, 200, 0), thickness=2, lineType=8, shift=0)
        frame1=cv2.line(img, pt1=(0, 240), pt2=(640, 240), color=(0, 200, 0), thickness=2, lineType=8, shift=0)

        #Smallframe= cv2.resize(img,(1280,960))
        cv2.imshow('detCam',frame1)
        cv2.moveWindow('detCam',0,0)
        item=""
        if cv2.waitKey(1)==ord('q'):
            break
    cam.release()
    cv2.destroyAllWindows()
```

APPENDIX C

Arduino code

```
#include <Wire.h>
#include <Adafruit_PWMServoDriver.h>
Adafruit_PWMServoDriver srituhobby = Adafruit_PWMServoDriver();

#define servoMIN 150
#define servoMAX 400


#define outputA 4  //CLock
#define outputB 5 //B
#define RotationMotorPos 8
#define RotationMotorNeg 9


#define in1 24 //Bmotor_pos
#define in2 25  //Bmotor_neg
#define in3 26 //Tmotor_pos
#define in4 27 //Tmotor_neg
#define Toplimit 22
#define Bottomlimit 23

#define position_sensor 3     //ir sensor


#define in5 7 //sliding_pos
#define in6 6 //sliding_neg
#define frontlimit 28
#define backlimit 29


#define lfr_in1 15  //orange
```

```
#define lfr_in2 16 //red
#define lfr_in3 17  //brown
#define lfr_in4 19  //black
#define lfr_enA 14  //yellow
#define lfr_enB 18  //white
#define lfr_trig1 31
#define lfr_echo1 30

int M1_Speed = 200; // speed of motor 1
int M2_Speed = 200; // speed of motor 2
int LeftRotationSpeed = 250;  // Left Rotation Speed
int RightRotationSpeed = 250; // Right Rotation Speed



int LEFT_SENSOR_0 ; //LEFT sensor (A0)
int LEFT_SENSOR_1  ; //LEFT middle sensor (A1)
int RIGHT_SENSOR_1 ;//RIGHT middle sensor (A2)
int RIGHT_SENSOR_0 ;//RIGHT sensor   (A3)
int RIGHT ; //RIGHT sensor (A5)
int LEFT ; //RIGHT sensor  (A4)




int counter = 0;
int aState;
int aLastState;

int veritcal_limit_count = 0;
long dur;
long dis;
long plant_range;
```

```
const int out = 10; //trigger
const int in = 11; //echo



byte servo1 = 1;
byte servo2 = 4;

int verpos1 = 400;
int Waterpump1 = 32, Waterpump2 = 33;
char buffer[16];






void setup()
{
  pinMode(in1, OUTPUT);
  pinMode(in2, OUTPUT);
  pinMode(in3, OUTPUT);
  pinMode(in4, OUTPUT);

  pinMode(Toplimit, INPUT);
  pinMode(Bottomlimit, INPUT);
  pinMode(LED_BUILTIN, OUTPUT);
  digitalWrite(in1, LOW);
  digitalWrite(in2, LOW);
  digitalWrite(in3, LOW);
  digitalWrite(in4, LOW);
  delay(2000);
```

```
pinMode (RotationMotorPos, OUTPUT);
pinMode (RotationMotorNeg, OUTPUT);
pinMode (position_sensor, INPUT);
digitalWrite(RotationMotorPos, LOW);
digitalWrite(RotationMotorNeg, LOW);
delayMicroseconds(1000);



pinMode(in5, OUTPUT);
pinMode(in6, OUTPUT);

pinMode(frontlimit, INPUT);
pinMode(backlimit, INPUT);
digitalWrite(in5, LOW);
digitalWrite(in6, LOW);
delay(1000);



pinMode (RotationMotorPos, OUTPUT);
pinMode (RotationMotorNeg, OUTPUT);
pinMode (outputB, INPUT);

// Reads the initial state of the outputA
//aLastState = digitalRead(outputA);





pinMode(in, INPUT);
pinMode(out, OUTPUT);
```

```
//veritcal_limit_count=0;
Serial.begin(9600);

pinMode(LED_BUILTIN, OUTPUT);
pinMode(lfr_in1, OUTPUT);
pinMode(lfr_in2, OUTPUT);
pinMode(lfr_in3, OUTPUT);
pinMode(lfr_in4, OUTPUT);
pinMode(lfr_trig1, OUTPUT);



pinMode(lfr_enA, OUTPUT);
pinMode(lfr_enB, OUTPUT);

pinMode(A0, INPUT); // initialize Left sensor as an input
pinMode(A1, INPUT); // initialize slight_left sensor as an input
pinMode(A2, INPUT); // initialize Left sensor as an input
pinMode(A3, INPUT); // initialize Right sensor as an input
pinMode(A4, INPUT); // initialize Left most sensor as an input
pinMode(A5, INPUT); // initialize Right most sensor as an input
pinMode(lfr_echo1, INPUT);
Serial.begin(9600);

srituhobby.begin();
srituhobby.setPWMFreq(60);
pinMode(Waterpump1, OUTPUT);
pinMode(Waterpump2, OUTPUT);
pinMode(LED_BUILTIN, OUTPUT);
digitalWrite(Waterpump1, LOW);
digitalWrite(Waterpump2, LOW);
digitalWrite(LED_BUILTIN, LOW);
while (!Serial)
{
    ; // wait for serial port to connect.
  }
}



long microsecondsToCentimeters(long microseconds)
{
  return microseconds / 29 / 2;
}


void WaterPump()
{
  if (Serial.available() > 0)
  {
    int size = Serial.readBytesUntil('\n', buffer, 12);
    Serial.println(buffer[0]);
    if (buffer[0] == 'S')
    {
      digitalWrite(LED_BUILTIN, HIGH);
      digitalWrite(Waterpump1, HIGH);
      digitalWrite(Waterpump2, LOW);
      delay(2000);
      digitalWrite(LED_BUILTIN, LOW);
      digitalWrite(Waterpump2, LOW);
      digitalWrite(Waterpump1, LOW);
      //delay(1000);
    }
  }
}
```

```
void plant_presence()
{
  digitalWrite(out, LOW);
  delayMicroseconds(2);
  digitalWrite(out, HIGH);
  delayMicroseconds(10);
  digitalWrite(out, LOW);
  dur = pulseIn(in, HIGH);
  plant_range = microsecondsToCentimeters(dur);
  Serial.println(String(plant_range));
  delay(1000);

}

void pest_detection()
{
  //srituhobby.setPWM(servo2, 0, verpos1);
  // Serial.println(servo2);
  //delay(500);
  //srituhobby.setPWM(servo1, 0, verpos1);
  //Serial.println(servo1);
  //delay(500);



   for (int pulse2 = 0; pulse2 < 275; pulse2++)
  {
    srituhobby.setPWM(servo2, 0, pulse2 );
    Serial.println(servo2);
    delay(10);
  }
  delay(1000);
```

```
for (int pulse = servoMIN; pulse < servoMAX; pulse++)
{
  WaterPump();
  srituhobby.setPWM(servo1, 0, pulse);
  Serial.println(servo1);
  delay(50);
}
delay(2000);


 for (int pulse2 = 0; pulse2 < 150; pulse2++)
{
  srituhobby.setPWM(servo2, 0, pulse2 );
  Serial.println(servo2);
  delay(10);
}
delay(1000);


for (int pulse = servoMAX; pulse > servoMIN; pulse--)
{
  WaterPump();
  srituhobby.setPWM(servo1, 0, pulse);
  Serial.println(servo1);
  delay(50);
}
delay(2000);

srituhobby.setPWM(servo2, 0, 212);
delay(1000);
srituhobby.setPWM(servo1, 0, 275);
```

```
  delay(1000);



}

void Counter()
{
  aState = digitalRead(outputA); // Reads the "current" state of the outputA
  // If the previous and the current state of the outputA are different, that means a
Pulse has occured
  if (aState != aLastState)
  {
    // If the outputB state is different to the outputA state, that means the encoder is
rotating clockwise
    if (digitalRead(outputB) != aState)
    {
      counter ++;
    }

    else
    {
      counter --;
    }
    Serial.print("Position: ");
    Serial.println(counter);
  }
  aLastState = aState; // Updates the previous state of the outputA with the current
state
}

void CW()   //45degrees
{
  while (counter != -11)
  {
    Counter();
    Serial.println(counter);
    digitalWrite(RotationMotorPos, HIGH);
    digitalWrite(RotationMotorNeg, LOW);
    delayMicroseconds(1000);
  }
}

void CW1()
{
  while (counter != -33)
  {
    Counter();
    digitalWrite(RotationMotorPos, HIGH);
    digitalWrite(RotationMotorNeg, LOW);
    delayMicroseconds(1000);
  }
}

void CCW_Zero_degree()    // 0 degree position
{
  while (counter != 0)
  {
    Counter();
    digitalWrite(RotationMotorPos, LOW);
    digitalWrite(RotationMotorNeg, HIGH);
    delayMicroseconds(1000);
  }
}



void CCW()
```

```
{
  while (counter != 11)
  {
    Counter();
    digitalWrite(RotationMotorPos, LOW);
    digitalWrite(RotationMotorNeg, HIGH);
    delayMicroseconds(1000);
  }
}

void CCW1()
{
  while (counter != 33)
  {
    Counter();
    digitalWrite(RotationMotorPos, LOW);
    digitalWrite(RotationMotorNeg, HIGH);
    delayMicroseconds(1000);
  }
}



void initial_pos_v()
{
  while (1)
  {
    if (digitalRead (Bottomlimit) == 0)
    {
      digitalWrite(in1, LOW);
      digitalWrite(in2, LOW);
      digitalWrite(in3, LOW);
      digitalWrite(in4, LOW);
      delay(2000);
      break;
    }
    digitalWrite(in1, LOW);
    digitalWrite(in2, HIGH);
    digitalWrite(in3, HIGH);
    digitalWrite(in4, LOW);
    delay(100);

  }
}

void armangle_initialization()
{
  while (1)
  {
    digitalWrite(RotationMotorPos, LOW);
    digitalWrite(RotationMotorNeg, HIGH);
    delayMicroseconds(1000);

    if (digitalRead(position_sensor) == 1)
    {
      digitalWrite(RotationMotorPos, LOW);
      digitalWrite(RotationMotorNeg, LOW);
      delay(1000);
      break;
    }

  }
```

```
}



void upward_movement()
{
  while (1)
  {
    plant_presence();
    if (veritcal_limit_count == 3)
    {
      digitalWrite(in1, LOW);
      digitalWrite(in2, LOW);
      digitalWrite(in3, LOW);
      digitalWrite(in4, LOW);
      delay(2000);
      break;
    }
    else if (plant_range > 35)
    {
      digitalWrite(in1, LOW);
      digitalWrite(in2, LOW);
      digitalWrite(in3, LOW);
      digitalWrite(in4, LOW);
      delay(2000);
      break;
    }

    else
    {
    digitalWrite(in1, HIGH);
    digitalWrite(in2, LOW);
    digitalWrite(in3, LOW);
    digitalWrite(in4, HIGH);
    delay(7500);
    digitalWrite(in1, LOW);
    digitalWrite(in2, LOW);
    digitalWrite(in3, LOW);
    digitalWrite(in4, LOW);
    delay(2000);
    veritcal_limit_count = veritcal_limit_count + 1;
    if(plant_range > 35)
    {
    pest_detection();
    delay(3000);
    }
    else
    {
      break;
    }
    //scanning part
    /*if(digitalRead (Toplimit) == 0)
      {
      digitalWrite(in1, LOW);
      digitalWrite(in2, LOW);
      digitalWrite(in3, LOW);
      digitalWrite(in4, LOW);
      delay(2000);
      break;
      }*/
    }
  }
}
```

```
void initial_pos_h()
{
  while (1)
  {
    digitalWrite(in5, LOW);
    digitalWrite(in6, HIGH);
    delay(100);
    if (digitalRead(backlimit) == 0)
    {

      digitalWrite(in5, LOW);
      digitalWrite(in6, LOW);
      delay(2000);
      break;
    }
  }
}

void final_pos_h()
{
  while (1)
  {
    digitalWrite(in5, HIGH);
    digitalWrite(in6, LOW);
    delay(100);
    if (digitalRead(frontlimit) == 0)
    {

      digitalWrite(in5, LOW);
      digitalWrite(in6, LOW);
      delay(2000);
      break;
    }
  }
}

void forward()
{
  digitalWrite(lfr_in1, HIGH);
  digitalWrite(lfr_in2, LOW);
  digitalWrite(lfr_in3, LOW);
  digitalWrite(lfr_in4, HIGH);

  analogWrite(lfr_enA, M1_Speed);
  analogWrite(lfr_enB, M2_Speed);
}

void backward()
{
  digitalWrite(lfr_in1, LOW);
  digitalWrite(lfr_in2, HIGH);
  digitalWrite(lfr_in3, LOW);
  digitalWrite(lfr_in4, HIGH);

  analogWrite(lfr_enA, M1_Speed);
  analogWrite(lfr_enB, M2_Speed);
}

void right()
{

  while (RIGHT != 0)
  {
    RIGHT = digitalRead(A5);
    digitalWrite(lfr_in1, LOW);
    digitalWrite(lfr_in2, LOW);
    digitalWrite(lfr_in3, LOW);
    digitalWrite(lfr_in4, HIGH);

    analogWrite(lfr_enA, LeftRotationSpeed);
    analogWrite(lfr_enB, RightRotationSpeed);

    delay(300);
  }
}

void slight_right()
{
  digitalWrite(lfr_in1, LOW);
  digitalWrite(lfr_in2, LOW);
  digitalWrite(lfr_in3, LOW);
  digitalWrite(lfr_in4, HIGH);

  analogWrite(lfr_enA, LeftRotationSpeed);
  analogWrite(lfr_enB, RightRotationSpeed);

}

void left()

{

  while (LEFT != 0)
  {

    LEFT = digitalRead(A4);
    digitalWrite(lfr_in1, HIGH);
    digitalWrite(lfr_in2, LOW);
    digitalWrite(lfr_in3, LOW);
    digitalWrite(lfr_in4, LOW);

    analogWrite(lfr_enA, LeftRotationSpeed);

    analogWrite(lfr_enB, RightRotationSpeed);
    delay(300);


  }
}

void slight_left()
{
  digitalWrite(lfr_in1, HIGH);
  digitalWrite(lfr_in2, LOW);
  digitalWrite(lfr_in3, LOW);
  digitalWrite(lfr_in4, LOW);

  analogWrite(lfr_enA, LeftRotationSpeed);

  analogWrite(lfr_enB, RightRotationSpeed);

}

void Stop()
{
  digitalWrite(lfr_in1, LOW);
  digitalWrite(lfr_in2, LOW);
  digitalWrite(lfr_in3, LOW);
  digitalWrite(lfr_in4, LOW);
}
```

```
void lfr()
{

  while (1)
  {
    digitalWrite(lfr_trig1, LOW);
    delay(2);
    digitalWrite(lfr_trig1, HIGH);
    delay(10);
    digitalWrite(lfr_trig1, LOW);
    long duration1 = pulseIn(lfr_echo1, HIGH);
    int cm1 = duration1 * 0.034 / 2;

    LEFT_SENSOR_0 = digitalRead(A0);  //LEFT sensor
    LEFT_SENSOR_1 = digitalRead(A1);  //LEFT middle sensor
    RIGHT_SENSOR_1 = digitalRead(A2); //RIGHT middle sensor
    RIGHT_SENSOR_0 = digitalRead(A3); //RIGHT sensor
    RIGHT = digitalRead(A5); //RIGHT sensor
    LEFT = digitalRead(A4); //RIGHT sensor

    if (cm1 >= 10 && cm1 <= 15)
    {
      digitalWrite(LED_BUILTIN, HIGH);
      Stop();
      Serial.print("Distance1: ");
      Serial.println(cm1);
      delay(3000);
      break;
    }

    else
    {
```

```
    if (LEFT_SENSOR_0 == 1 && LEFT_SENSOR_1 == 1 &&
RIGHT_SENSOR_1 == 1 && RIGHT_SENSOR_0 == 1 && RIGHT == 0 &&
LEFT == 0)
    {
      forward(); //FORWARD
    }
    if (LEFT_SENSOR_0 == 1 && LEFT_SENSOR_1 == 1 &&
RIGHT_SENSOR_1 == 1 && RIGHT_SENSOR_0 == 1 && RIGHT == 1 &&
LEFT == 1)
    {
      forward(); //FORWARD
    }

    else if (LEFT_SENSOR_0 == 0 && LEFT_SENSOR_1 == 1 &&
RIGHT_SENSOR_1 == 1 && RIGHT_SENSOR_0 == 1 && RIGHT == 0 &&
LEFT == 0)
    {
      slight_right();
    }

    else if (LEFT_SENSOR_0 == 0 && LEFT_SENSOR_1 == 1 &&
RIGHT_SENSOR_1 == 1 && RIGHT_SENSOR_0 == 1 && RIGHT == 1 &&
LEFT == 0)
    {
      slight_right();
    }

    else if (LEFT_SENSOR_0 == 1 && LEFT_SENSOR_1 == 1 &&
RIGHT_SENSOR_1 == 1 && RIGHT_SENSOR_0 == 0 && RIGHT == 0 &&
LEFT == 0)
    {
      slight_left(); //Move Left
    }
    else if (LEFT_SENSOR_0 == 1 && LEFT_SENSOR_1 == 1 &&
RIGHT_SENSOR_1 == 1 && RIGHT_SENSOR_0 == 0 && RIGHT == 0 &&
LEFT == 1)
    {
     slight_left(); //Move Left
    }

    else if (LEFT_SENSOR_0 == 1 && LEFT_SENSOR_1 == 1 &&
RIGHT_SENSOR_1 == 1 && RIGHT_SENSOR_0 == 1 && RIGHT == 1 &&
LEFT == 0)
    {
     right();
    }

    /* else if(LEFT_SENSOR_0==0 && LEFT_SENSOR_1==0 &&
RIGHT_SENSOR_1==0 && RIGHT_SENSOR_0==1)
     {
      right();
     }
    */
    else if (LEFT_SENSOR_0 == 1 && LEFT_SENSOR_1 == 1 &&
RIGHT_SENSOR_1 == 1 && RIGHT_SENSOR_0 == 1 && RIGHT == 0 &&
LEFT == 1)
    {
     left();
    }



    /* else if(LEFT_SENSOR_0==1 && LEFT_SENSOR_1==0 &&
RIGHT_SENSOR_1==0 && RIGHT_SENSOR_0==0)
     {
       left();
       }
     */
     else if (LEFT_SENSOR_0 == 0 && LEFT_SENSOR_1 == 0 &&
RIGHT_SENSOR_1 == 0 && RIGHT_SENSOR_0 == 0 && RIGHT == 0 &&
LEFT == 0)
     {
       Stop();
     }

   }
  }



}



void loop()
{

  initial_pos_v();
  Serial.println("Vertical postion set");
  delay(1000);
  armangle_initialization();
  Serial.println("Arm angle set");
  delay(1000);
  srituhobby.setPWM(servo2, 0, 212);
  delay(1000);
  srituhobby.setPWM(servo1, 0, 275);
  delay(3000);
  initial_pos_h();
  Serial.println("horizontal postion set");
```

```
delay(3000);



lfr();
delay(1000);

counter = 0;
aLastState = digitalRead(outputA);
Counter();
CW();
digitalWrite(RotationMotorPos, LOW);
digitalWrite(RotationMotorNeg, LOW);
Serial.println("45_degree postion set");
delay(3000);
final_pos_h();
delay(3000);
plant_presence();
if (plant_range < 35)
{
  pest_detection();
  veritcal_limit_count = 0;
upward_movement();   //also consists of scanning part
delay(3000);
Serial.println("upward movement set");

}
//scan here also



initial_pos_h();
delay(3000);
initial_pos_v();
delay(3000);
```

```
  //CCW_Zero_degree();
  armangle_initialization();
  digitalWrite(RotationMotorPos, LOW);
  digitalWrite(RotationMotorNeg, LOW);
  Serial.println("0_degree postion set");
  // armangle_initialization();
  delay(3000);

  while(1)
{
 digitalWrite(lfr_trig1, LOW);
    delay(2);
    digitalWrite(lfr_trig1, HIGH);
    delay(10);
    digitalWrite(lfr_trig1, LOW);
    long duration1 = pulseIn(lfr_echo1, HIGH);
    int cm1 = duration1 * 0.034 / 2;

 if(cm1>25)
 {
  digitalWrite(lfr_in1, LOW);
  digitalWrite(lfr_in2, LOW);
  digitalWrite(lfr_in3, LOW);
  digitalWrite(lfr_in4, LOW);
  delay(100);
  break;
    }

  digitalWrite(lfr_in1, HIGH);
  digitalWrite(lfr_in2, LOW);
  digitalWrite(lfr_in3, LOW);
  digitalWrite(lfr_in4, HIGH);
  analogWrite(lfr_enA, M1_Speed);
  analogWrite(lfr_enB, M2_Speed);
  }

  //digitalWrite(lfr_in1, HIGH);
  //digitalWrite(lfr_in2, LOW);
  //digitalWrite(lfr_in3, LOW);
  //digitalWrite(lfr_in4, HIGH);

  //analogWrite(lfr_enA, M1_Speed);
  //analogWrite(lfr_enB, M2_Speed);

  //delay(1500);



  //digitalWrite(lfr_in1, LOW);
  //digitalWrite(lfr_in2, LOW);
  //digitalWrite(lfr_in3, LOW);
  //digitalWrite(lfr_in4, LOW);

  lfr();
  delay(3000);

 counter = 0;
  aLastState = digitalRead(outputA);
  Counter();

  CW1();
  digitalWrite(RotationMotorPos, LOW);
  digitalWrite(RotationMotorNeg, LOW);
  Serial.println("315_degree postion set");
  delay(3000);
```

```
  final_pos_h();
  delay(3000);

  plant_presence();
  if (plant_range < 35)
  {
    pest_detection();
    veritcal_limit_count = 0;
  upward_movement();      //also consists of scanning part
  delay(3000);
  Serial.println("upward movement set");

  }
  //scan here also

  initial_pos_h();
  delay(3000);
  initial_pos_v();
  delay(3000);

  //CCW_Zero_degree();
  armangle_initialization();
  digitalWrite(RotationMotorPos, LOW);
  digitalWrite(RotationMotorNeg, LOW);
  Serial.println("0_degree postion set");
  delay(3000);
  //armangle_initialization();

  while(1)
{
 digitalWrite(lfr_trig1, LOW);
    delay(2);
    digitalWrite(lfr_trig1, HIGH);
    delay(10);
    digitalWrite(lfr_trig1, LOW);
    long duration1 = pulseIn(lfr_echo1, HIGH);
    int cm1 = duration1 * 0.034 / 2;

 if(cm1>25)
 {
  digitalWrite(lfr_in1, LOW);
  digitalWrite(lfr_in2, LOW);
  digitalWrite(lfr_in3, LOW);
  digitalWrite(lfr_in4, LOW);
  delay(100);
  break;
    }

  digitalWrite(lfr_in1, HIGH);
  digitalWrite(lfr_in2, LOW);
  digitalWrite(lfr_in3, LOW);
  digitalWrite(lfr_in4, HIGH);

  analogWrite(lfr_enA, M1_Speed);
  analogWrite(lfr_enB, M2_Speed);
  }



    //digitalWrite(lfr_in1, HIGH);
  //digitalWrite(lfr_in2, LOW);
  //digitalWrite(lfr_in3, LOW);
  //digitalWrite(lfr_in4, HIGH);

  //analogWrite(lfr_enA, M1_Speed);
  //analogWrite(lfr_enB, M2_Speed);
  //delay(1500);



  //digitalWrite(lfr_in1, LOW);
  //digitalWrite(lfr_in2, LOW);
  //digitalWrite(lfr_in3, LOW);
  //digitalWrite(lfr_in4, LOW);

 lfr();
  // 50% plant scanning completion



 }
```

Pest Detection

ORIGINALITY REPORT

15%	15%	4%	11%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	Submitted to Higher Education Commission Pakistan Student Paper	3%
2	github.com Internet Source	2%
3	Submitted to SUNY, Empire State College Student Paper	1%
4	amsdottorato.unibo.it Internet Source	1%
5	ckk.com.tr Internet Source	1%
6	www.coursehero.com Internet Source	1%
7	tnsroindia.org.in Internet Source	1%
8	wikimili.com Internet Source	1%
9	Submitted to SVKM International School Student Paper	1%

10	towardsdatascience.com Internet Source	1%
11	Submitted to University of Southern Mindanao Student Paper	<1%
12	Submitted to Segi University College Student Paper	<1%
13	Submitted to South Bank University Student Paper	<1%
14	d-nb.info Internet Source	<1%
15	ro.uow.edu.au Internet Source	<1%
16	www.hindawi.com Internet Source	<1%
17	www.bgdi.org Internet Source	<1%
18	Divya Arora, Mehak Garg, Megha Gupta. "Diving deep in Deep Convolutional Neural Network", 2020 2nd International Conference on Advances in Computing, Communication Control and Networking (ICACCCN), 2020 Publication	<1%
19	doaj.org Internet Source	<1%

20	doctorpenguin.com Internet Source	<1%
21	oudernaturally.com Internet Source	<1%
22	tudr.thapar.edu:8080 Internet Source	<1%
23	Nitharshana Mahenthiran, Haarini Sittampalam, Sinthumai Yogarajah, Sathurshana Jeyarajah et al. "Smart Pest Management: An Augmented Reality-Based Approach for an Organic Cultivation", 2021 2nd International Informatics and Software Engineering Conference (IISEC), 2021 Publication	<1%
24	dash.harvard.Edu Internet Source	<1%
25	"Proceedings of Second International Conference on Advances in Computer Engineering and Communication Systems", Springer Science and Business Media LLC, 2022 Publication	<1%
26	deepai.org Internet Source	<1%
27	eprints.utar.edu.my Internet Source	<1%

28	erepo.uef.fi Internet Source	<1 %
29	norma.ncirl.ie Internet Source	<1 %
30	www.igi-global.com Internet Source	<1 %
31	"Advances in Computational Intelligence", Springer Science and Business Media LLC, 2019 Publication	<1 %

Exclude quotes	Off	Exclude matches	Off
Exclude bibliography	On		

Real Time Pest Detection and Fumigation Using Machine Learning

Syeda Briha Haider, Burhanuddin Zohair and Noor Ul Ain Ayaz

Department of Electrical Engineering
Bahria University Karachi Campus
75260, Pakistan

{barihahaider, burhan.nicobar.bz & noorulainayaz1}@gmail.com

***Abstract*—** *Pakistan generates 70% economy from agricultural sector. It is the largest growing sector in Pakistan. It contributes 24% to the GDP. The geographical location and environmental conditions of Pakistan are suitable for the growth of most of the crops. Every crop requires different climatic condition to grow. Crops such as rice, cotton, vegetables and fruits are mainly exported to Europe and other countries. These countries face harsh climate which does not satisfy the necessary climatic conditions required for several crops and therefore these countries export these crops from countries like Pakistan due to its excellent crop quality, production and growth rate. The crop demand is directly dependant on the crop quality which is a major factor that must not be compromised. Providing the necessary conditions to grow and protecting from any diseases is the major goal that needs to be achieved. These crops are prone to get affected by any diseases that can be caused by different pests. Pests largely compromise the crop quality and destroy several crops which affects the economy. Therefore a pesticide needs to be used in order to get rid of any pests present on the crops that might compromise the crop quality. Large amounts of plants may contain several pests of different sizes and colors which may not be visible to human eye. Therefore, we have designed a robotic arm that can move around the crop field and effectively scan the plants for the presence of any pests on the crops and spraying pesticide to get rid of them. The pesticide used will get rid of these pests without causing any harm to the nature as it is environmental friendly. Using Innovative technology we will try to reduce wastage of crops, improve crop quality and reducing poverty. This will reduce labour work and promote increased crop demand resulting in good impact on the economy*

Index Terms*—Pest , poverty and GDP.*

I. Introduction

Agriculture is the largest and fastest growing sector in Pakistan. It is a large contributor to the economy of Pakistan, contributes about 24% to the (Gross Domestic Product) GDP. Large part of the population is depended on Agricultural sector. It provides employment to the labors, food for the population as well as means of foreign exchange earnings. The geographical location of Pakistan makes it suitable for the growth of most of the crops and therefore other countries living in harsh environments need to import these crops from countries like Pakistan.

The demand is directly related to the quality of these crops grown. The quality of the crops should be remarkable in order to export these crops. Several factors are responsible for compromising the crop quality. One major factor is the issue of Pests. Pests are creatures that can potentially destroy and damage the crops which would compromise the quality and therefore resulting in the crops not being exported. This results in huge amount of crops being wasted and affecting the economy. To counter this problem, Computer vision and Machine learning can be used to locate and identify multiples pests present on the crops and getting rid of them by fumigating the plant using pesticide. The idea behind the project is to create a robotic arm that can move in vertical farms to scan any pests present that may not be visible to human eye and get rid of them by spraying pesticide. Robots are replacing manpower in every industrial sector. The main advantage of using robots is that they will help to increase the productivity level and production rate by an appreciable amount. They will be providing the crop with the precise amount of improved production. Moreover, it will be taking the load of the labour shortage and not a single square inch of the land will go unobservant as the robot will ensure crop production at every corner of the fenced land.

II. Design and Implementation

A. Applying the concepts of AI and machine learning, the Jetson Nano has been trained to use pre-trained networks. It uses transfer learning which a method of training pretrained network is by transferring knowledge of previously trained network to new task. Neural networks are a type of machine learning technique that help mimic like human brain in order to perform object detection, recognition and segmentation. Installing necessary libraries and packages will enable the device to be trained on our dataset. . Once the device is trained it can detect the presence of the pests. To make the project movable LFR principle is used which will enable the robotic arm to follow a certain path. Horizontal, vertical and rotational movements are adjusted every time a plant is nearby which will be detected using ultrasonic sensors. Using Jetson Nano and Arduino as main controllers in conjunction with dc motors, servo motors IR sensors, rotatory encoders and Logitech c270 webcam as main components of the design. All the plants are scanned and once the presence of pests is detected motor is activated to spray pesticide on the pests to get rid of them. The arm keeps moving until all plants are scanned and pests are killed.

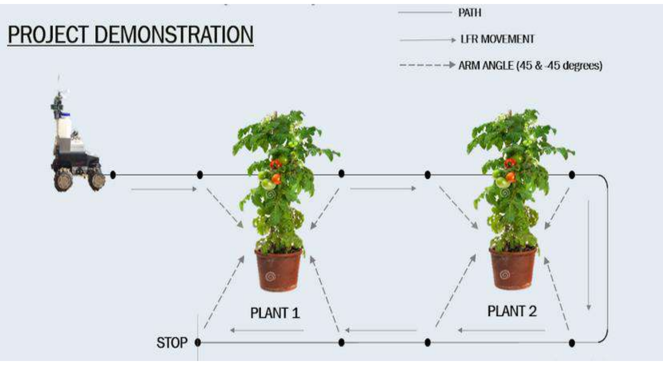


Figure 1: conceptual design

Machine learning techniques

CNN – CONVOLUTION NEURAL NETWORK

Convolutional Neural Networks (CNNs) that have brought perfection to this field over the time. They are widely used type of neural networks used in computer vision for recognition and detection of objects .CNN's takes an input image assign importance to some objects and differentiate one form from another. CNN requires the least amount of pre-processing compared to the algorithm. CNN architecture in analogous to the patterns of neurons in human brain. They use convolution phenomena to detect edges from an image. CNN has the following layers:

Convolution Layers, pooling layers and fully connected layers. Within a convolutional layer, the input is first transformed and then passed to the next layer. It uses filter to transform data. A filter is simply a matrix of randomized number values. These layers work together to adapt special features of an image.

SSD – SINGLE SHOT DETECTOR

To train the dataset we have used SSD MobileNet which is a Convolution Neural Network architecture widely used for object detection. SSD stands for Single Shot Detector which has a single convolution network it uses a base architecture called mobileNet (which are efficient convolution neural networks) and has several convolution layers. SSD takes only one shot in order to detect several objects in an image while some others need more than 1 shot which makes it much faster. It has a feed-forward convolution network, which keeps producing boxes and scores around desired objects. It extracts feature map and the apply convolution in order to detect any objects present.

B. Algorithm design

Keeping in mind the problem statement and objectives presented in chapter one, a rough sketch of the pest detecting robot is presented in the form of a flow chart in order to specify clearly what the Robot will do and how the sequence of operations will take place in the overall process.

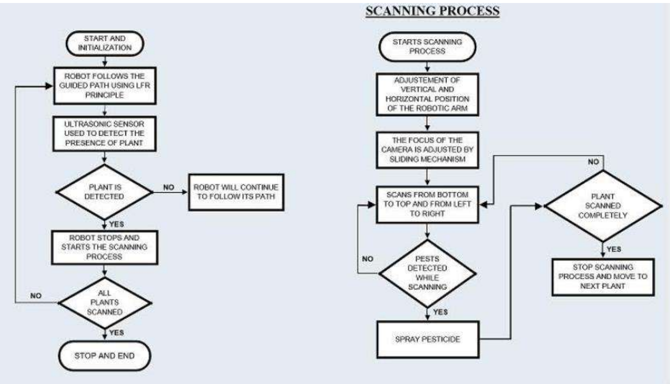

START AND INITIALIZATION
ROBOT FOLLOWS THE GUIDED PATH USING LFR PRINCIPLE
ULTRASONIC SENSOR USED TO DETECT THE PRESENCE OF PLANT
PLANT IS DETECTED
NO
ROBOT WILL CONTINUE TO FOLLOW ITS PATH
YES
ROBOT STOPS AND STARTS THE SCANNING PROCESS
ALL PLANTS SCANNED
NO
YES
STOP AND END
SCANNING PROCESS
STARTS SCANNING PROCESS
ADJUSTEMENT OF VERTICAL AND HORIZONTAL POSITION OF THE ROBOTIC ARM
THE FOCUS OF THE CAMERA IS ADJUSTED BY SLIDING MECHANISM
SCANS FROM BOTTOM TO TOP AND FROM LEFT TO RIGHT
NO
PESTS DETECTED WHILE SCANNING
YES
SPRAY PESTICIDE
PLANT SCANNED COMPLETELY
NO
YES
STOP SCANNING PROCESS AND MOVE TO NEXT PLANT

Fig. 2. Design Flow

The Robot is autonomous once powered on it will move around a plant to cover the whole plant for scanning from bottom to top. For this scanning purpose we've fitted a Logitech c270 webcam through a sliding mechanism. This sliding mechanism is used to adjust the camera focus for pests that would be detect by a sensor. For vertical movement of arm a sliding rod has been used which is attached to the DC motors. As the motor rotates the rod will rotate and the arm will move vertically. The arm will stop at desired instances as set in the algorithm. Limits switches are used to set limit for the vertical movement. At each instance the arms stops vertically, the rotational movement is put into action. The gear is connected to the motor. The motor rotation results in gear rotation, which will rotate the bearing and hence the arm will rotate. The The bearing and gears are attached together by a belt. The main component used for this part are rotatory encoders which counts the steps the arms takes to rotate. It is set to stop after certain steps and as it stops horizontal movement of the arm is put into action. It uses a sliding mechanism which is mainly to set the focus of the camera attached to it. The camera's tilt and pan movement is controlled using servo motors.When it stops the camera starts detecting and sprays pesticide if pests are detected if not it continues to move to next step.

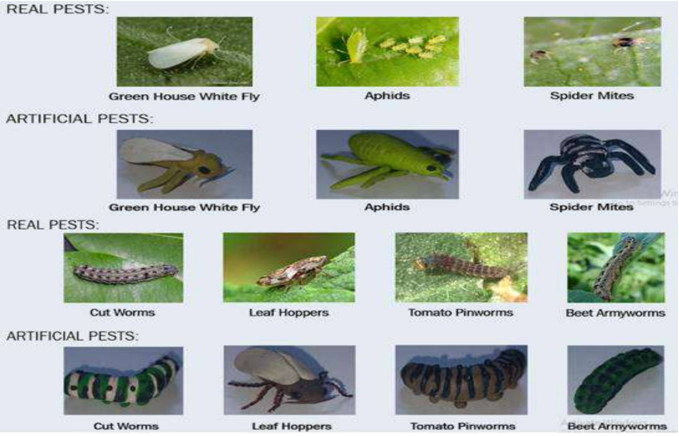


Fig. 3. Dataset

C. Robot design

The CAD design was implemented for line follower robot and for robotic arm while keeping the functionalities of robot to perform the detection and spraying in mind.

The designed was implemented on AUTOCAD software and the screen-shot of the structure are as follows:

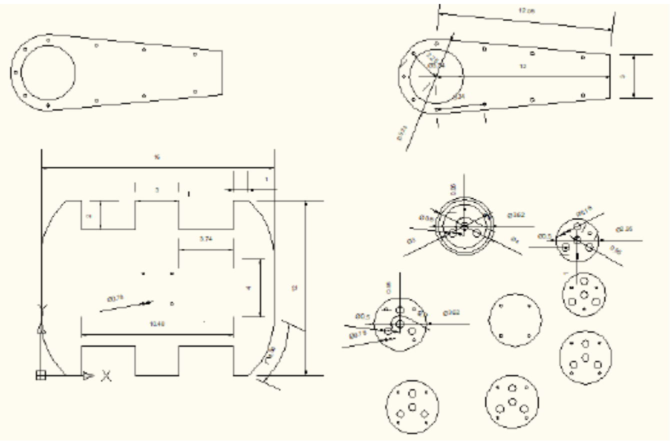


Fig. 4. AUTOCAD Design; Body (Left), arm (Up), Supporting plates (Right)

D. Mechanical Structure

The robotic arm is designed in such a way that a camera is attached to it along with controllers and motors that will work simultaneously to detect and kill pests. The hardware is designed in a way to move around the field and also move horizontally, vertically and rotationally for proper scanning of the plant

- Line Following Robot:

The robotic arm moves on the phenomena of LFR which uses IR sensors to follow a certain path and stop when an obstacle is detected which will be sensed by ultrasonic sensors .

- Vertical Movement

The vertical movement of the robotic arm aims to adjust the height of the arm according to the plant height. DC motors are used along with limit switches to set the limit and move the arm vertically.

- Rotational Movement

The gear is connected to the DC motor. The motor rotation results in gear rotation, which will rotate the bearing and hence the arm will rotate. The bearing and gears are attached together by a belt. Rotatory encoders are used which counts the steps the arms takes to rotate. The algorithm is designed to make the arm stop at an angle of 45° (11 steps) and -45° (22 steps).

- Sliding Mechanism

The sliding mechanism of the robotic arm is designed to set the focus of the camera. Using Dc motors the rod will move and will extend the camera attached to it 6” away from the plant as this distance results in a good focus of the camera. Ultrasonic sensor will measure the distance and adjust accordingly.

- Camera Movement

Using servo motors to tilt (move vertically) and pan (move horizontally) the camera.

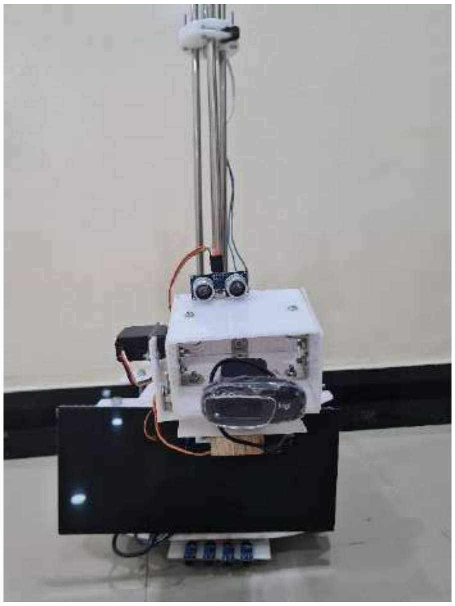


Fig. 5. Autonomous pest detecting robot

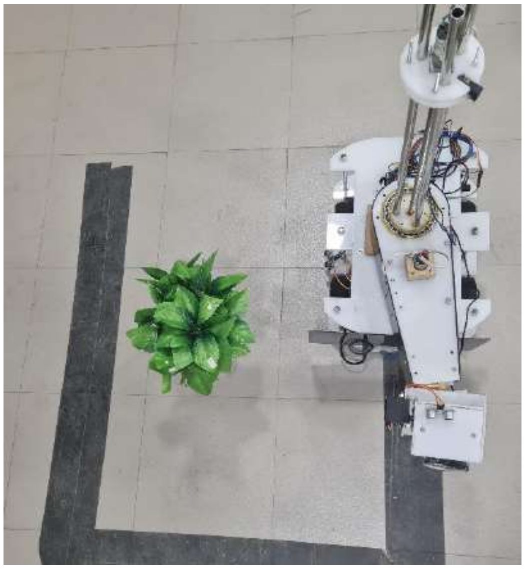
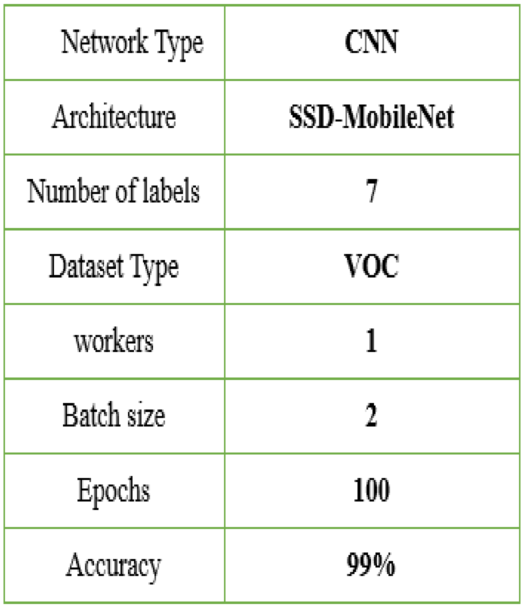


Fig. 6. Robot while following track

III. Results



Network Type	CNN
Architecture	SSD-MobileNet
Number of labels	7
Dataset Type	VOC
workers	1
Batch size	2
Epochs	100
Accuracy	99%

Fig. 7. Training Parameters and Accuracy

A. Detection Results

Our dataset consisted of 1000 pictures and it was trained for 100 epochs. After training the results were observed. Bounding boxes were drawn over detected objects which also mentioned the name of the pests. It shows that the Greenhouse whitefly was detected at an accuracy of 98%, tomato pinworm at an accuracy 68.5% and cutworms with an accuracy of 99.9 % as illustrated in the image below.

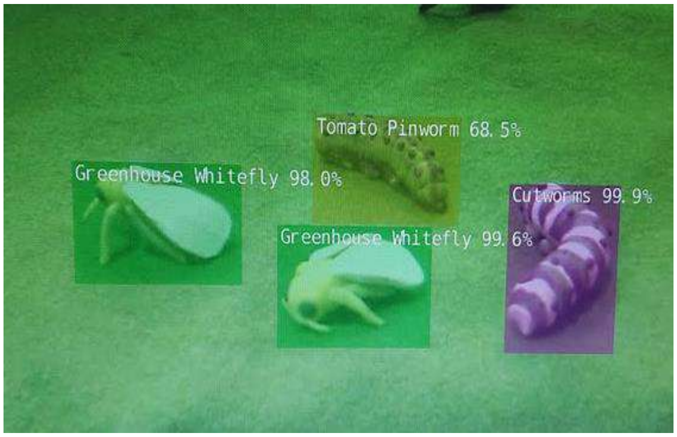


Figure 8: Pest detection results with accuracy

B. Accuracy graphs with respect to distance between plant and Robotic arm

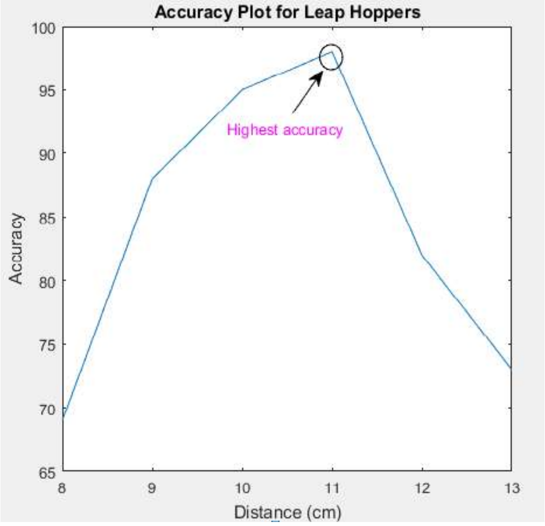


Figure 9: Accuracy Plot for Leaf Hoppers

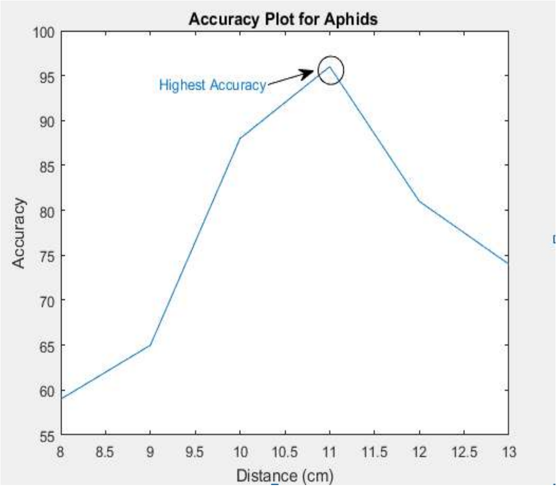


Figure 10: Accuracy Plot for Aphids

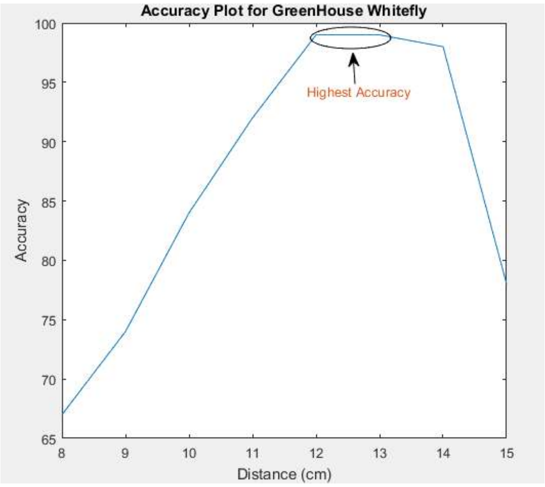


Figure 11: Accuracy Plot for GreenHouse Whitefly

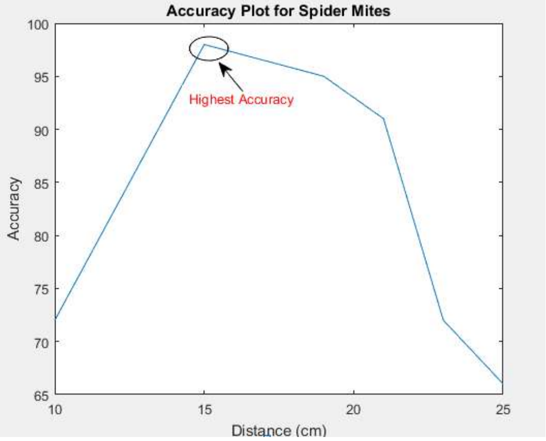


Figure 12: Accuracy Plot for Spider Mites

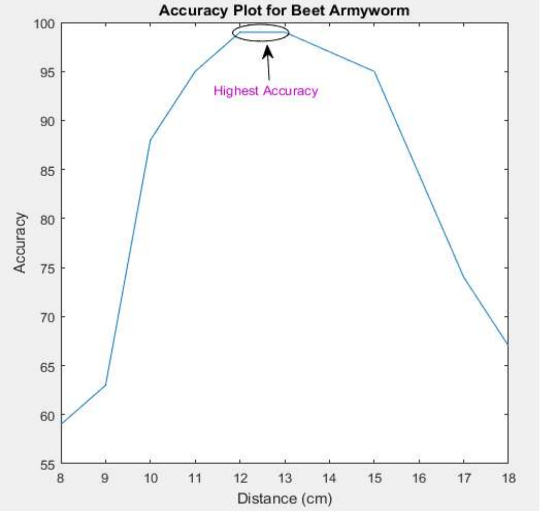


Figure 13: Accuracy Plot for Beet Armyworm

C. Accuracy bar chart with respect to epochs

The Bar graph demonstrate the accuracy with respect to epochs (training cycles).We have come up with the result that best accuracy has been achieved at 100 epochs than 30 epochs.

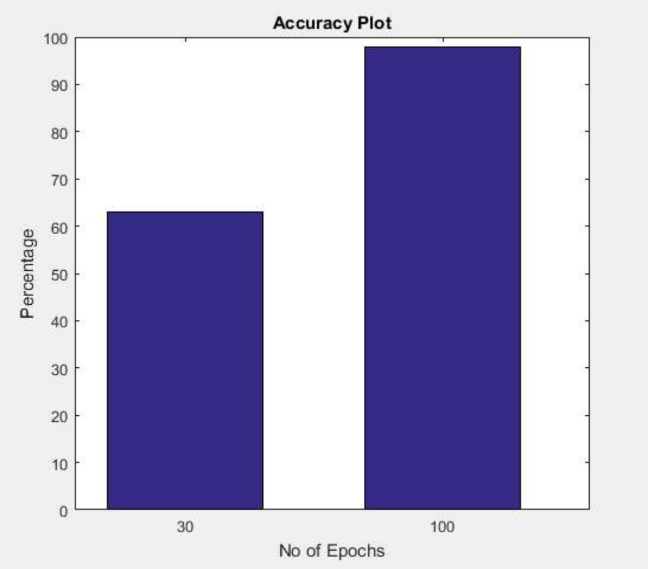


Figure 14: Percentage Accuracy Bar Chart

IV. Conclusion

Agricultural sector is the largest contributor to Pakistan's economy. There are several environmental factors that can inhibit or destroy the growth of plants and compromise crop quality such as pests. Poor quality can lead to low demand and largely affect the economy. To counter this problem, Computer vision and Machine learning can be used to detect multiple pests present and then to spray pesticide on those specific parts of the plants on which the pests are present. The Robotic Arm is based on LFR principle to move around fields and scan all plants. Applying the concepts of AI and machine learning, the Jetson Nano has been trained to use pretrained networks through transfer learning to recognize.The dataset has been collected using the camera capture tool and nano has been trained on the dataset. Using SSD-MobileNet as the CNN architecture which gives best results and training on 100 epochs. 99% accuracy was achieved in pest detection.

References

[1] Kasinathan, Thenmozhi, Dakshayani Singaraju, and Srinivasulu Reddy Uyyala. "Insect classification and detection in field crops using modern machine learning techniques." *Information Processing in Agriculture* 8.3 (2021): 446-457.

[2] Yang, Jiudong, and Jianping Li. "Application of deep convolution neural network." *2017 14th International Computer Conference on Wavelet Active Media Technology and Information Processing (ICCWAMTIP)*. IEEE, 2017.

[3] Younis, Ayesha, et al. "Real-time object detection using pre-trained deep learning models MobileNet-SSD." *Proceedings of 2020 the 6th international conference on computing and data engineering*. 2020.

[4] Mahesh, Batta. "Machine learning algorithms-a review." *International Journal of Science and Research (IJSR).[Internet]* 9 (2020): 381-386.

[5] B. Jayanthi, T. A. Priyanka, V. B. Shalini and R. K. Grace, "Pest Detection in Crops Using Deep Neural Networks," 2022 8th International Conference on Advanced Computing and Communication Systems (ICACCS), 2022, pp. 29-32, doi: 10.1109/ICACCS54159.2022.9785155.

[6] Liu, Jun, and Xuewei Wang. "Plant diseases and pests detection based on deep learning: a review." *Plant Methods* 17.1 (2021): 1-18.

Reserach Paper

ORIGINALITY REPORT

3%	2%	1%	1%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	towardsdatascience.com Internet Source	1%
2	Submitted to University College London Student Paper	1%
3	www.ijraset.com Internet Source	<1%
4	erepo.uef.fi Internet Source	<1%

Exclude quotes	Off	Exclude matches	Off
Exclude bibliography	On		