A Major Project Report On

WEED DETECTION AND ITS MINIMISATION IN COTTON CROP

Submitted in partial fulfilment of the requirements for the

Award of the degree of

BACHELOR OF TECHNOLOGY

IN

ELECTRONICS AND INSTRUMENTATION ENGINEERING

By

G. B. V. SUBRAHMANYAM	188W1A1014
P. GOPI	**188W1A1036**
M. SHASI KUMAR	188W1A1030
S. VENKATESH	178W1A1046

Under the guidance of

Mr. P. SRINIVAS M Tech (PhD)

DEPARTMENT OF ELECTRONICS AND INSTRUMENTATION ENGINEERING

V.R. SIDDHARTHA ENGINEERING COLLEGE
(AUTONOMOUS)

(Affiliated to JNTUK, Kakinada)

Sponsored by SAGTE, Kanuru, Vijayawada-520 007

(Approved by AICTE, Accredited by NBA and NAAC A GRADE)

2021-2022

DEPARTMENT OF ELECTRONICS AND INSTRUMENTATION ENGINEERING

V.R. SIDDHARTHA ENGINEERING COLLEGE(Autonomous): Vijayawada-7

CERTIFICATE

This is to certify that the Major Project titled "WEED DETECTION AND ITS MINIMISATION IN COTTON CROP" is a bonafide record of work done by G. B. V. Subrahmanyam (188W1A1014), P. Gopi (188W1A1036) M. Sashi Kumar (188W1A1030), S. Venkatesh (178W1A1046) under my guidance and supervision and is submitted in partial fulfilment of the requirements for the award of the degree of Bachelor of Technology in Electronics & Instrumentation Engineering, V.R. Siddhartha Engineering College, (Autonomous, Affiliated to JNTUK, Kakinada) during the academic year 2021-2022.

Guide: P. SRINIVAS	(Dr. G. N. SWAMY)
MTech (PhD)	PhD
Associate Professor	Professor & Head
Dept. of EIE	Dept. of EIE

ACKNOWLEDGEMENT

First and foremost, we would like to express our special gratitude and thanks to **Venkateswarulu** farmer of Cotton field at Jupudi, Ibrahimpatnam for giving the information and valuable time to us.

We express our profound gratitude to our esteemed guide, **Sri P. SRINIVAS**, associate professor for his guidance during the major project.

We are deeply indebted to **Dr. G. N. SWAMY**, professor and head of the department of electronics and instrumentation engineering, for his cooperation given throughout the major project for the facilities provided.

We express our profound gratitude to our beloved principal **Dr. A. V. RATNA PRASAD** and our management for providing library facilities for completing our major project successfully.

By

G. B. V. Subrahmanyam

P. Gopi

M. Shasi Kumar

S. Venktasesh

ABSTRACT

Weed identification in vegetable plantation is more challenging than crop weed identification due to their random plant spacing. So far, little work has been found on identifying weeds in vegetable plantation. Traditional methods of crop weed identification used to be mainly focused on identifying weed directly; however, there is a large variation in weed species.

Weeds compete with agricultural crops for plant nutrients and water and are one of the most significant sources of pests and diseases. Effective control of weeds is critical for maximising moisture storage and crop yields, reducing the weed seed bank, and meeting quality standards at harvest. It is very important to prevent weeds from setting seed, as a single parent can produce hundreds to thousands of weeds. Many species also have an inherent dormancy period, so seeds persist in the soil seed-bank. These offspring can be a problem not only in the next season, but for many future seasons.

CONTENTS

List of Figures

CHAPTER 1

HERBICIDE SPRAYING ROVER

1.1 Introduction

Weed control is crucial to agriculture, because weeds decrease yields, increase production costs, interfere with harvest, and lower product quality. Weeds also impede irrigation water-flow, interfere with pesticide application, and harbour disease organisms. Weed management can take up a significant proportion of pre-harvest variable costs in vegetable production.

Weed management commences prior to planting of the crop and does not stop until the crop has been harvested and residual weeds are destroyed or cultivated. Early methods of weed control included mowing, flooding, smothering, burning, and crop rotation. Though these methods are still important, other means are perhaps more typical today, particularly the use of herbicide chemicals. Another technique is to introduce insects that attack only the unwanted plant and destroy it while leaving the crop plants unharmed

1.2 Motivation

In cotton, weeds cause several direct and indirect negative impacts, such as reducing fibre quality and reducing crop yield etc., Therefore, for effective weed management in cotton, growers should concentrate their efforts on weed management in the early part of the growing season. In order to collect more data about weed removal techniques we gone through some cotton fields, and we've interacted with the farmers for the challenges they are facing.

On the objective we've met three farmers namely Subbarao, Srinu, Venkateswarulu at Jupudi Village on 9th-Febraury-2021.After listening to their words, we've noticed some complications of the farmers. After the seed plantation, when the seed at leaf area and canopy development the plant is so sensitive to the pesticide than the weed. Usually, they will cover the plants and sprays the entire field, to protect of plant from pesticide. The figure 1.1 shows how weed grows around crop.

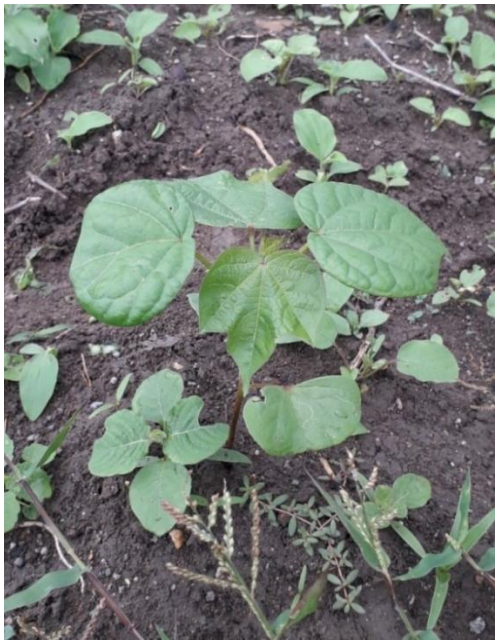


Figure 1.1 Weed Near the Cotton Crop

So, it is very difficult to spray the pesticide for the entire field by covering each plant and is time taking process. The pesticide can cause health issues to the labour. The cost for spraying the entire field also exorbitant nearly 1000Rs per acer and a worker for spraying the entire field charges at least 200RS. Totally it costs nearly 1200Rs per acer.

Other than that, by employing labour for that is also a burdensome process. Weed detection was done by inspecting each and every place in the field, then weeds were removed manually. It requires 6 to 7 members, nearly 200Rs per head and about 5 to 6 hours to remove all the weed from an acer. By this method, it costs nearly 1200 to 1400Rs per acer. The removal of weed should be taken place 2 to 3 times per a crop.

On the whole, it costs 4000Rs to 5000Rs for a crop and the two methods should be implemented since the cotton is very sensitive one.

1.3 Problem Description

- The cost for spraying of pesticide for the entire field is exorbitant, and can cause health issues to the labour.
- Other than that, by employing labour for that process is time consuming, it also lavishes.
- Both the methods used are the expensive and time taking processes.

Chapter 2

LITERATURE SURVEY

Umamaheswari S [1] proposed a paper where classification is done by constructing bounding box and the coordinates are saved in JSON file. More than one rectangle can be stored for each image which indicates the regions have the desired object and the machine learns the features from them. The feature extraction module which consists of several modules and layers of Convolutional Neural Network is implemented by using Googlenet-Overfeat model in TensorFlow framework. Maged Wafy [2] present the results of applying Scale Invariant Feature Transform (SIFT) for the extraction of interest point from image. SIFT has Scale-space extrema detection, Key points localization, Orientation assignment, Keypoint descriptor, Matching. The algorithm was applied to detect species of weed seeds mixed with wheat grains in samples.

C. Thirumarai Selvi's [4] work starts with, the given RGB images are converted into Gray scale images during pre-processing step. Various noises and unwanted background objects or images are suppressed using filtering techniques. Weed features are properly analysed and extracted using feature extraction process and groups the weed and crops separately in classification procedure. Convolutional Neural Network technique is used for feature extraction. Classifiers are trained first, then validated and finally tested for many images using artificial neural network. validated and tested using images of different weed some classifiers are artificial neural network.

Siddhesh Badhan's [5] work, In this paper, initially the reconstruction of the 3D model of crops from the video of the crops and then detection and classification of weeds and crops will be done. The 3D reconstruction consists of capturing video followed by frame extraction (OpenCV), feature matching (SFIT), sparse and dense re-construction (Visual SFM). After 3D reconstruction is performed, the next part is detecting the weeds in the crops. The weed detection part mainly consists of Data acquisition, Pre-processing, Image segmentation, Training the model (ML & DL based), Testing the model and real-time weed detection in crops.

Adnan Farooq [6] The main contribution of this paper is: CNN is introduced for spectral-spatial analysis on hyperspectral image for weed classification. The methodology of this can be decomposed into three steps: 1) hyperspectral data collection using VA210

filter and JAI BM-141 camera; 2) generating labelled samples of 2000 image patches, for each weed category, and 3) classification is done by CNN architecture and the HoG method. M.Dian Bah [7] proposed with the Hough transform, this method detects crop rows using the skeletons. Once the crop rows are identified, image (UAV) is segmented in superpixels. In this paper, features are extracted from the pretrained Residual Network with 18 layers (ResNet18).

Oscar Barrero's [8] proposal is to acquire the images of the field, a CMOS camera upon an autonomous electrical delta wind plane Phantom FX-61 by Hobby King is used. Here Pixhawk with ardupilot and Mission Planner as software is used for control. And then Pix4Dmapper Pro software is used to generate an ortho mosaic map of the field. In Knowledge Database Construction; Grey Level Co-occurrence Matrix (GLCM) for texture discrimination and Normalized Difference Index (NDI) for colour discrimination. Finally, ANN is used for the training and feature extraction.

Aichen Wang [9] In this work, an encoder-decoder deep learning network was investigated for pixel-wise semantic segmentation of crop and weed. The model trained with NIR images yielded much better result, with MIoU value of 79.28%. The network was implemented relying on the Google TensorFlow library with the programming language Python 3.5. Transfer learning for a convolutional neural network that consists of convolution base and fully-connected layers at the end, means to retrain the final layers of the network with new training data based on a previously trained network. The encoder-decoder network based on a pretrained model on PASCAL VOC 2012 dataset from VOC challenges with 11530 images, leads to much less computation load and training data.

Inkyu Sa, Zetao Chen [10] Here the CNN-based dense semantic classification is implemented for weed detection with aerial multispectral images taken from an MAV. The encoder-decoder cascaded deep neural network is trained on a dataset. Sequoia multispectral sensor is used for image acquisition, the annotated images are fed into SegNet, Firstly, the frequency of appearance (FoA) for each class is adapted based on our training dataset for better class balancing. This is used to weigh each class inside the neural network loss function, and then a simple input/output layer is implemented that reads images and outputs them to the subsequent concatenation layer, which is useful for

hyperspectral image processing. Finally, the model was embedded that can be carried by a small MAV.

Hari Shankar RL [11] A camera, interfaced with the main controller on the quadcopter, the Ardupilot, which is mounted on a UAV. The images that are taken by this camera is transmitted by ardupilot to the main system, where the images are processed by frame by the OpenCV using python programming language. The nodeMCU and serial protocol MAVlinks are installed for transmission of data from the controller using Wi-Fi. In the training phase, dataset related to images are given in prior to the SVM for comparison of images. Using these data, a default model is generated which is used for comparison and generate results. After distinguishing the green segment pattern from the crops and the weeds, the presence of weed can be determined using clustering.

Vitali Czymmek [12] In this two cameras Nikon DSLR and Basler daA1920 are used as sensors. The dataset of 10,000 images were used for training purpose. A forked Windows version of the open-source DL framework Darknet is used for implementation. For displaying the training results the open-source framework OpenCV was used. To speed up the training we used the GPU-acceleration library cu DNN. The YOLO approach is used for object detection, The network was trained with the previously noted dataset for 55.000 iterations resulting in the loss function, and it detected the multiple weed species in organic farming.

The block diagram of the Spraying rover is discussed in the next chapter.

CHAPTER 3

BLOCK DIAGRAM OF HERBICIDE SPRAY ROVER

The approach of the method is, all the leaves are not in the same shape. There is a lot of difference in the shape of cotton leaves when compared to the weed. The image acquired is from the crop field. The image passes through various stages of processing. Initially, image pre-processing is performed to suppress unwanted distortions and to enhance some image features important for further processing. The Block Diagram of the entire process is shown in the Fig.2.1.

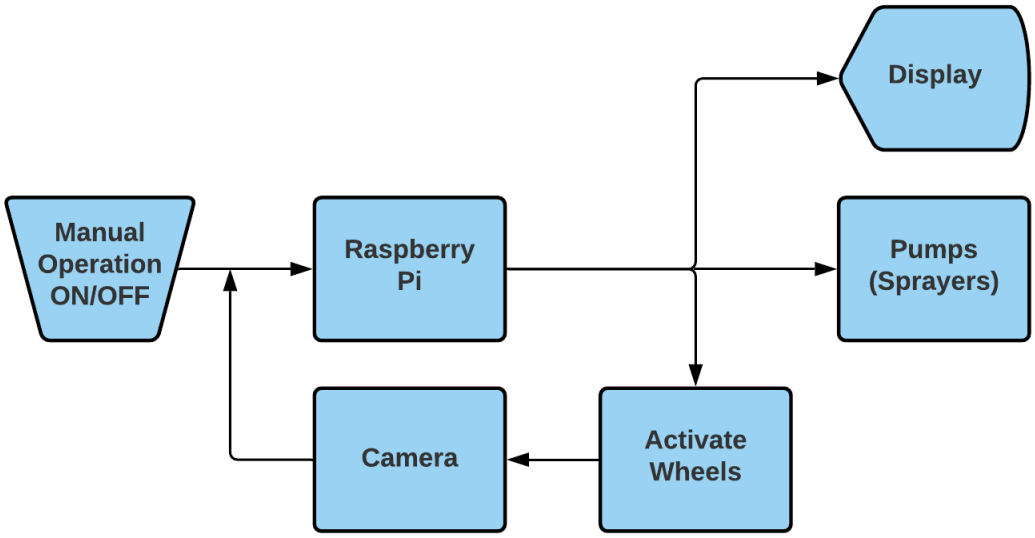


Figure 2.1 Block Diagram of Herbicide Spray Rover

From the image, contours of the entire plant region are obtained and this gives the corresponding coordinate values. These values are then used to draw rectangles bounding the contour areas and numbering is given as well. Bigger ones are the crop and smaller ones are weeds as per the assumptions considered. After recognizing the position of weed in the image plane, the signal for activation of the sprayer is given. The whole process repeats over the entire length of the row where the crops are cultivated.

The implementation of this approach is shown in the flowchart figure2.2 For implementing the weed detection approach, a forked Windows version of the open-source deep learning framework Darknet is used. Total layers in this approach are roughly 106 and 32 filters. The YOLO network was trained with the previously noted dataset for

nearly 4000 images of cotton leaves from the open-source site of Kaggle. The YOLO approach reframes object detection as a single regression problem from image pixels to bounding box coordinates.

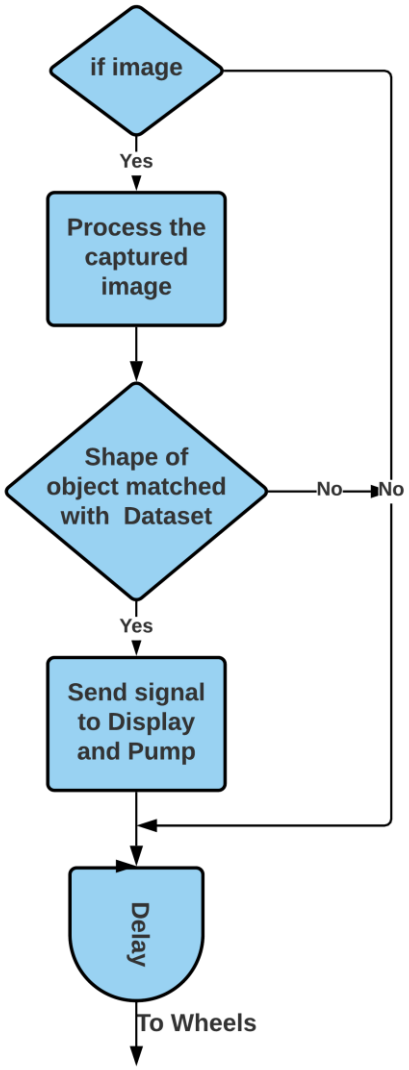


Figure 2.2 Flow of Software Execution inside Raspberry Pi

The captured images are compared with the trained images and classified whether it is a cotton or not. If not, it is considered as the weed. By recognizing the coordinates of weed and respective sprayer will gets energized to spray the pesticide on it.

Figure 2.3 Herbicide Spraying Rover

The entire setup including pesticide tank for spraying, cam for capturing the field, along with a power source for the driving all the components, is arranged in a small rover with four wheels as shown in figure 2.3.

The hardware used in the model is explained in the next chapter in detail

CHAPTER 3

HARDWARE COMPONENTS

3.1. Raspberry Pi

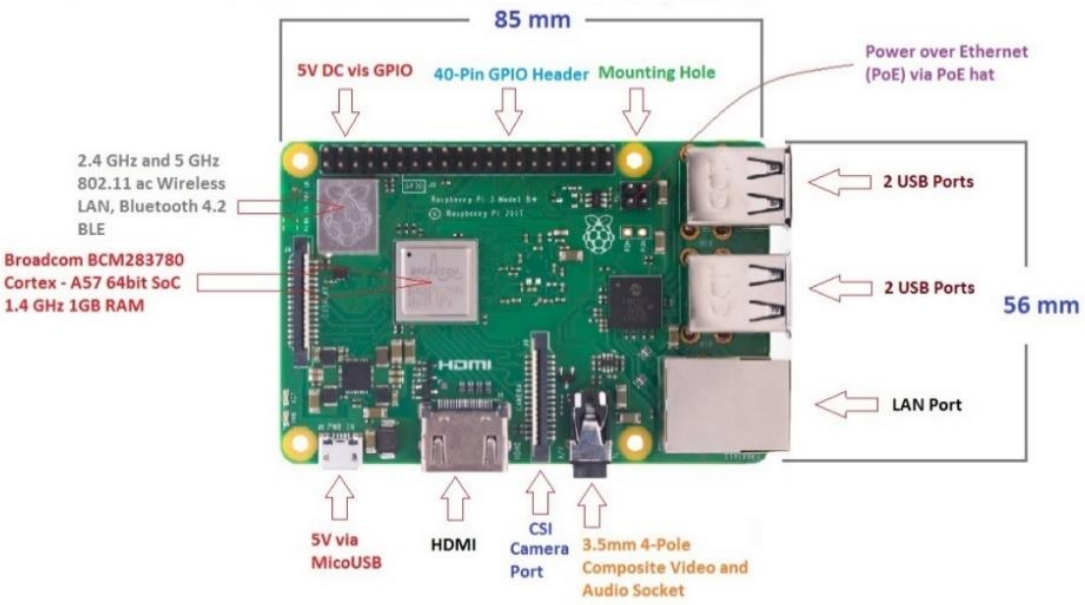


Figure 3.1 Raspberry Pi Board

The Raspberry Pi Foundation, a UK non-profit that strives to educate people in computing and make computing education more accessible, has created a series of single-board computers known as the Raspberry Pi. Figure 3.1 depicts a typical Raspberry Pi diagram.

The Raspberry Pi was first published in 2012, and since then, various revisions and modifications have been developed. The original Pi had a single-core 700MHz CPU and just 256MB RAM, while the most recent model has a quad-core 1.5GHz CPU and 4GB RAM.

People use the Raspberry Pi all across the world to study programming, develop hardware projects, automate their homes, deploy Kubernetes clusters and Edge computing, and even employ them in industrial applications.

The Raspberry Pi is a low-cost computer that runs Linux and has a set of GPIO (general purpose input/output) ports for controlling electronic components and experimenting with the Internet of Things (IoT).

Components on the Pi:

Once Raspberry Pi is chosen it's important to learn a few things about it, starting with a look at the different parts on the PCB and what each part does.

3.1.1. Processor / SoC (System on Chip):

A Broadcom BCM2835 System on Chip module is included with the Raspberry Pi. The processor is an ARM1176JZF-S. [13]

The Raspberry Pi's Broadcom SoC is comparable to a smartphone chip (Android or iPhone). The Raspberry Pi has a real-world performance of about 0.041 GFLOPS while running at 700 MHz by default. The Raspberry Pi's CPU capability is comparable to a 300 MHz Pentium II from 1997-1999, while the GPU offers 1 Gpixel/s, 1.5 Gtexel/s, or 24 GFLOPS of general-purpose work, and the Raspberry Pi's graphics capabilities are about equivalent to the Xbox of 2001.

3.1.2. Power source:

The Raspberry Pi is a gadget that uses 700mA (3W) of electricity. A Micro USB charger or the GPIO header provide power. Any decent smartphone charger will enough to keep the Pi running.

3.1.3. SD Card:

Onboard storage is not accessible on the Raspberry Pi. The operating system is installed on an SD card that is placed into the Raspberry Pi's SD card port. Any computer with a card reader may load the operating system onto the card.

3.1.4. GPIO:

GPIO – General Purpose Input Output

A general-purpose input/output (GPIO) pin on an integrated circuit can be controlled by the user at run time, including whether it is an input or output pin.

GPIO capabilities may include:

- GPIO pins can be configured to be input or output
- GPIO pins can be enabled/disabled
- Input values are readable (typically high=1, low=0)

- Output values are writable/readable
- Input values can often be used as IRQs (typically for wakeup events)

The creation A 26-pin 2.54 mm (100 mil) extension header, labelled P1, is located in a 2x13 strip on the Raspberry Pi board. They have 8 GPIO pins, as well as I2C, SPI, and UART connections, as well as +3.3 V, +5 V, and GND supply lines. The first pin in the first column and on the bottom row is pin one. Figure 3.2 depicts the pin description.

Pin#	NAME		NAME	Pin#
01	3.3v DC Power		DC Power 5v	02
03	GPIO02 (SDA1 , I²C)		DC Power 5v	04
05	GPIO03 (SCL1 , I²C)		Ground	06
07	GPIO04 (GPIO_GCLK)		(TXD0) GPIO14	08
09	Ground		(RXD0) GPIO15	10
11	GPIO17 (GPIO_GEN0)		(GPIO_GEN1) GPIO18	12
13	GPIO27 (GPIO_GEN2)		Ground	14
15	GPIO22 (GPIO_GEN3)		(GPIO_GEN4) GPIO23	16
17	3.3v DC Power		(GPIO_GEN5) GPIO24	18
19	GPIO10 (SPI_MOSI)		Ground	20
21	GPIO09 (SPI_MISO)		(GPIO_GEN6) GPIO25	22
23	GPIO11 (SPI_CLK)		(SPI_CE0_N) GPIO08	24
25	Ground		(SPI_CE1_N) GPIO07	26
27	ID_SD (I²C ID EEPROM)		(I²C ID EEPROM) ID_SC	28
29	GPIO05		Ground	30
31	GPIO06		GPIO12	32
33	GPIO13		Ground	34
35	GPIO19		GPIO16	36
37	GPIO26		GPIO20	38
39	Ground		GPIO21	40

Figure 3.2 Pin Configuration of Raspberry Pi

3.1.5. DSI Connector:

The Mobile Industry Processor Interface (MIPI) Alliance's Display Serial Interface (DSI) protocol aims to reduce the cost of display controllers in mobile devices. LCD and comparable display technologies are frequently attacked. It specifies a serial bus and a communication protocol between the host (image data source) and the device (destination of the image data). The DSI connection may be used to connect a DSI compliant LCD screen, albeit extra drivers may be required to run the display.

3.1.6. RCA Video:

RCA On all Raspberry Pi versions, video outputs (PAL and NTSC) are accessible. The Rpi may be linked to any television or screen that has an RCA connection.

3.1.7. Audio Jack:

For stereo audio output, the Rpi has a standard 3.5 mm TRS connection. Any 3.5mm audio cable or headphone may be immediately attached. Although this jack cannot accept audio input, USB microphones or sound cards can be utilised.

3.1.8. Status LEDs:

There are 5 status LEDs on the Raspberry pi as in the figure 3.3 that show the status of various activities as follows

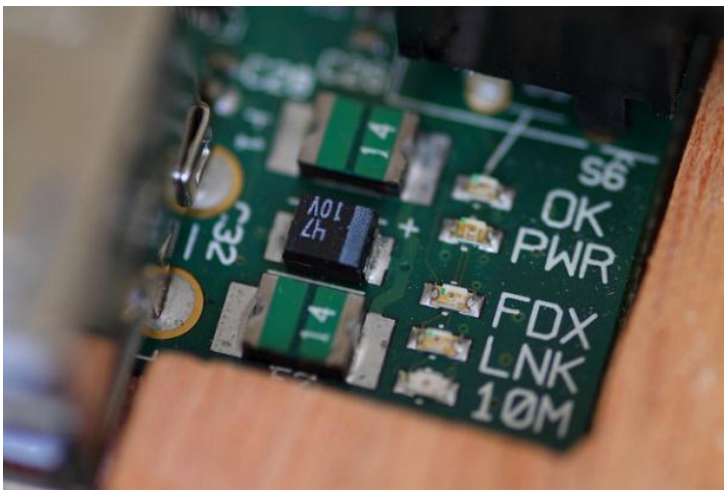


Figure 3.3 Status LEDs

"OK"	SD Card Access (via GPIO16) – labelled as "OK" on Model B Rev1.0 boards and "ACT" on Model B Rev2.0 and Model A boards
"POWER"	3.3 V Power – labelled as "PWR" on all boards
"FDX"	Full Duplex (LAN) **(Model B)** – labelled as "FDX" on all boards
"LNK"	Link/Activity (LAN) **(Model B)** – labelled as "LNK" on all boards
"10M/100"	10/100Mbit (LAN) **(Model B)** – labelled (incorrectly) as "10M" on Model B Rev1.0 boards and "100" on Model B Rev2.0 and Model A boards

3.1.9. USB 2.0 Port:

USB 2.0 ports are used to attach peripherals to the Raspberry Pi, such as a mouse or keyboard. Model A has one port, Model B has two, and Model B+ has four. An externally powered USB hub, which is available as a standard Pi accessory, can expand the number of ports.

3.1.10. Ethernet

On Model B and B+, there is an Ethernet port. Using the Ethernet connector, it may be linked to a network or the internet. Microchip's LAN9512 LAN controller chip is in charge of the Ethernet ports.

3.1.11. CSI connector:

Camera Serial Interface (CSI) is a serial interface created by the MIPI (Mobile Industry Processor Interface) consortium to connect digital cameras to mobile processors. The Raspberry Pi Foundation offers a camera designed specifically for the Pi that can be attached to the Pi through the CSI connection.

3.1.12. JTAG headers:

JTAG stands for 'Joint Test Action Group,' which was founded in the mid-1980s to address test point access concerns on PCBs containing surface mount devices. The TAP was created by the organisation to provide access to device pins via a serial connection (Test Access Port). The approach was accepted as an international standard in 1990. (IEEE Std 1149.1). This standardised connector is currently included as a feature on thousands of devices, allowing test and design engineers to access pins.

3.1.13. HDMI:

HDMI – High-Definition Multimedia Interface

HDMI 1.3 a type A port is provided on the RPi to connect with HDMI screens.

3.2. Pi Camera

Figure 3.4 5MP Pi Camera

The Raspberry Pi 5MP camera board is a custom-designed camera board with a flexible ribbon wire as in the figure 3.4 that allows it to work with Raspberry Pi devices. A fixed lens with a resolution of 5 megapixels is built into the camera board. With a resolution of 2592 x 1944 pixels, this camera board can capture stunning moments and record high-quality films in 1080p @ 30fps, 720p @ 60fps, and 640x480p 60/90 format.

3.3. 4-Channel Relay Module

The four-channel relay module has four 5V relays as well as the accompanying switching and isolating components, allowing for an easy interface with a microcontroller or sensor with the fewest possible components and connections. The contacts on each relay are rated for 250VAC, 30VDC, and 10A, respectively, as indicated on the body of the relays. The pin description is seen in table 3.1 below.

Table 3.1 Pin Description of the 4-Channel Relay Module

Pin Number	Pin Name	Description
1	GND	Ground reference for the module
2	IN1	Input to activate relay 1
3	IN2	Input to activate relay 2
4	IN3	Input to activate relay 3
5	IN4	Input to activate relay 4
6	V_{CC}	Power supply for the relay module
7	V_{CC}	Power supply selection jumper
8	JD-V_{CC}	Alternate power pin for the relay module

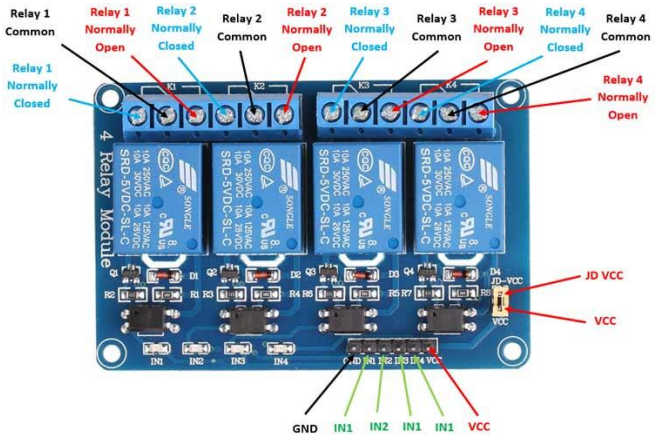


Figure 3.5 4-Channel Relay Module

3.4. 5V DC Motor Pumps

Figure 3.6 shows the 5V DC motor, is a low-cost mini submersible type water pump that works on 3-6V DC. It is extremely simple and easy to use. Just immerse the pump in water, connect a suitable pipe to the outlet and power the motor with 3-6V to start pumping water. Great for building science projects, fire-extinguishers, firefighting robots, fountains, waterfalls, plant watering systems etc.

Figure 3.6 5V DC Motor Pump

This motor is small, compact and light. It can be controlled from a micro controller/Arduino using our DC Motor Drivers or one of our Relay Boards. You may use our 5V SMPS Power Supply Adapter to run this pump. You may also use our 6V Solar Panel to run the pump with appropriate a 6V voltage regulator.

3.5. L239 Driver IC and DC Motors

The L293 is quadruple high-current half-H driver, is designed to provide bidirectional drive currents of up to 1 A at voltages from 4.5 V to 36 V. The Driver IC is as shown in figure8. The device is designed to drive inductive loads such as relays,

solenoids, dc and bipolar stepping motors, as well as other high-current/high-voltage loads in positive-supply applications.

Figure 3.7 L239 Driver IC and DC Motors

When an enable input is high, the associated drivers are enabled, and their outputs are active and in phase with their inputs. When the enable input is low, those drivers are disabled, and their outputs are off and in the high-impedance state. With the proper data inputs, each pair of drivers forms a full-H (or bridge) reversible drive suitable for solenoid or motor applications. Figure 3.7 shows the driver IC and DC motor

Centre Shaft Economy Series DC Motor is high-quality low-cost DC geared motor is shown in Figure8. It has steel gears and pinions to ensure longer life and better wear and tear properties. The gears are fixed on hardened steel spindles polished to a mirror finish. The output shaft rotates in a plastic bushing. The whole assembly is covered with a plastic ring. Gearbox is sealed and lubricated with lithium.

How these parts are interfaced to the Raspberry Pi is discussed in the very next chapter.

CHAPTER 4

HARDWARE INTERFACING

4.1. Interfacing Pi Camera

The Pi Camera module is a camera which can be used to take pictures and high-definition video. Raspberry Pi Board has CSI (Camera Serial Interface) interface to which we can attach the Pi Camera module directly. This Pi Camera module can attach to the Raspberry Pi's CSI port using a 15-pin ribbon cable as shown in the figure 4.1

Connect Pi Camera to CSI interface of Raspberry Pi board as shown below

Figure 4.1 Interfacing Pi Camera to Raspberry Pi

To use Pi Camera python-based library we have to include it in our program as given below

>>import picamera

This picamera library has PiCamera class for the camera module.

Python program for Video Recording:

Start recording using pi camera

>>camera.start_recording("/home/pi/demo.h264") []

wait for video to record

>>camera.wait_recording(20)

stop recording

>>camera.stop_recording()

>>camera.close()

The video will be saved into the folder that we have given in the statement.

4.2. Interfacing Relay Module

Let's start by importing the GPIO and time modules, which is the first step in this code. I used the time module to add a pause between scripts in this case.

>>import RPi.GPIO as GPIO

Connect the power supply to the common terminal and the NO node to the one LED to operate a single relay in the module (Just for testing) as in the figure 4.2. To control the relay, choose and configure a GPIO pin. The Relay is then controlled using the Python code below.

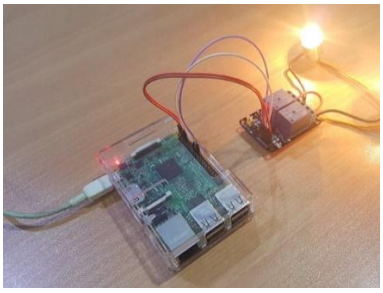


Figure 4.2 Interfacing Relay Module to Raspberry Pi

>>GPIO.output(Relay1_GPIO, GPIO.LOW)

>>sleep(1)

>>GPIO.output(Relay1_GPIO, GPIO.HIGH)

>>sleep(1)

Now for the proposed purpose there is a need for 3 channels and it can be achieved by a 4-Channel Relay Module and the remaining one is unused.

And the connections are made as per the given snippet below: That is, the three input pins are connected to the 17, 3,4 pins of RPi.

>>GPIO.output(17,False)

>>GPIO.output(3,False)

>>GPIO.output(4,False) // since there are three sprayers

4.3. Powering the Pumps through the Relay Module

The pumps used for spraying use the supply of 12V DC, but the RPi is not capable of producing that much output through the GPIO, So, in order to supply enough power to the pumps, introducing the Relays.

Connected the common terminals of the relays with 12V DC supply and all the three NO pins to 17, 3, and 4 pins of RPi.

Hence, achieved the control of pumps with RPi through Relay Module

The power supply could be through batteries or by choosing a suitable adopter for converting 230V AC TO 12V DC.

4.4. Interfacing L293d driver IC

The circuit for operating a dc motor using a raspberry pi is fairly as simple as in figure 4.3 to create. To begin, connect the l293d's pins 8 and 16 (vcc2 and vcc1) to an external 5v supply (assuming you are using a 5v motor).

On the l293d, there are four ground pins. Connect pin 4 to the supply's gnd. Also, connect the l293d's ground pin to the raspberry pi's gnd pin.

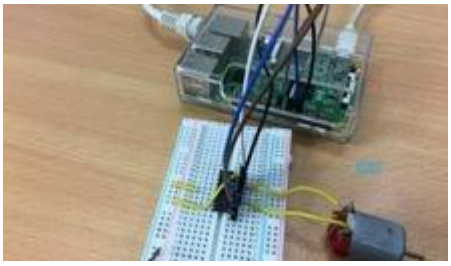


Figure 4.3 Interfacing L293d driver IC

Finally, the enable and control input pins are present. Connect pin 1 of the l293d (1,2en) to the raspberry pi's GPIO25 (physical pin 22). Then, for control input pins 2 and 7 (1a and 2a, respectively), connect them to GPIO24 (physical pin 18) and GPIO23 (physical pin 16).

Note: All the pins set up in this interfacing are not distinct we can use any GPIO pins for this

>>GPIO.OUTPUT(23,FALSE)

>>GPIO.OUTPUT(24,FALSE)

>>GPIO.OUTPUT(25,FALSE)

By using these statements, driver IC controls the DC Motor

In this model to run the wheels of the rover, the pins 26, 19, 27, 13 GPIO to run the two DC motors

The technique used in the detection process and the software used is discussed in the next chapter.

CHAPTER 5

SOFTWARE

5.1. Convolutional Neural Networks

A Convolutional Neural Network (ConvNet/CNN) is a Deep Learning algorithm that takes an input picture, assigns relevance (learnable weights and biases) to various aspects/objects within the image, and then separates them. A ConvNet requires substantially less pre-processing than other classification algorithms. ConvNets can search and locate these filters/characteristics with enough training, whereas filters are hand-engineered in primitive techniques. [14]

Convolutional Neural Networks (ConvNet/CNN) are Deep Learning algorithms that take an input image and give significance (learnable weights and biases) to various aspects/objects within the image before separating them. Other classification techniques need far more pre-processing than a ConvNet. With adequate training, ConvNets can search for and discover these filters/characteristics, whereas filters are hand-engineered using primitive methodologies.

Layers of CNN:

CNN has many layers as in the figure 5.1, and are explained in the sub chapters.

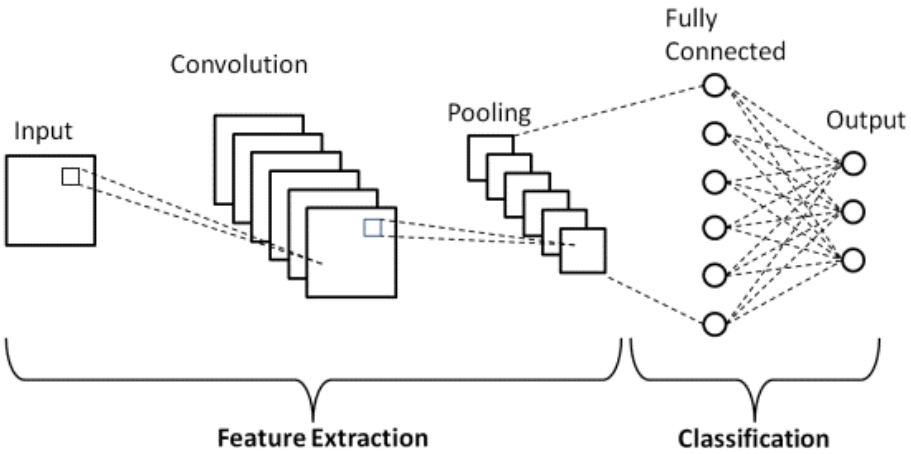


Figure 5.1 Architecture of CNN

i. Input layer
ii. Convo layer (convolution+Relu)

iii. Pooling layer

iv. Fully connected (FC) layer

v. Softmax /logistic layer

vi. Output layer

5.1.1 Input Layer

Image data should be present in the CNN input layer. Image data is represented using a three-dimensional matrix. It has to be converted from a matrix to a single column. You must transform a picture with dimensions of 28 28 = 784 to 784 x 1 before putting it into the input. If you have "n" training samples, the input dimension will be (784, n)

5.1.2 Convolution Layer

It is the first stage in gathering usable information from an image. Many filters in the convolution layer execute the convolution technique. Every image may be conceived of as a pixel value matrix. Consider a 5x5 picture with pixel values of 0 or 1- A three-dimensional filter matrix is also given as demonstrated in figure5.2. Slide the filter matrix over the picture and compute the dot product to obtain the convolved feature matrix.

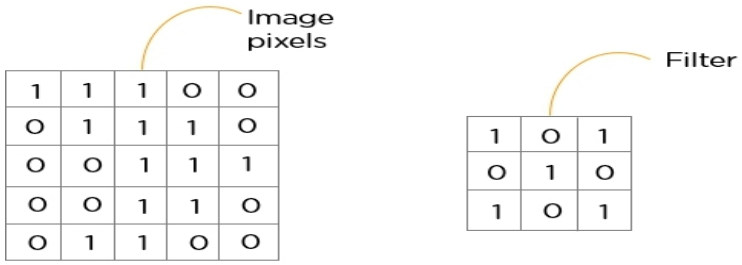


Figure 5.2 Convolution Layer

5.1.3 Activation Layer

The rectified linear unit is abbreviated as ReLU. Once all of the feature maps have been retrieved, they must be moved to a ReLU activation layer.

ReLU conducts an element-by-element process before setting all negative(-ve) pixels to 0. It adds nonlinearity to the network, and the resulting output is a corrected feature map. Figure 5.3 depicts the graph of a ReLU activation function.

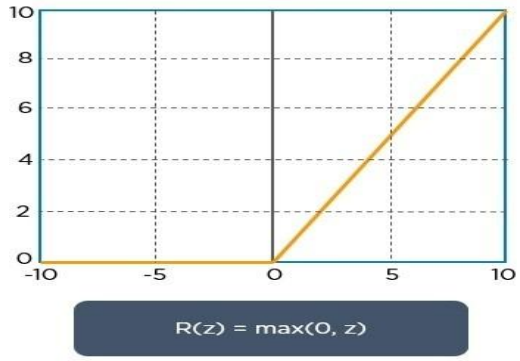


Figure 5.3 ReLU Function

5.1.4 Pooling Layer

Pooling is one of the downsampling methods used to reduce the dimensionality of a feature map. The corrected feature map is now passed through one pooling layer, resulting in a pooled feature map, which is then down sampled and output. Figure 5.4 shows how the pooling is done.

The pooling layer uses a number of filters and kernels to identify various features of the picture as needed, such as edges, corners, body, feathers, and eyes.

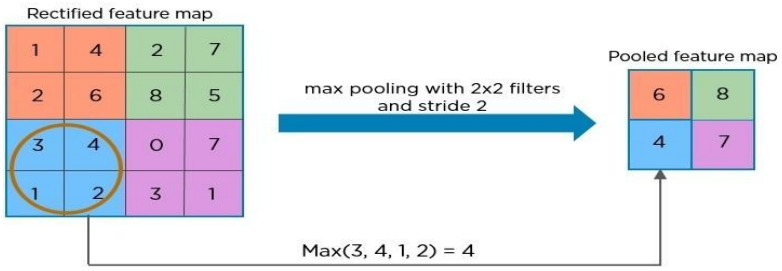


Figure 5.4 2X2 Filter for Pooling

5.1.5 Flatten Layer

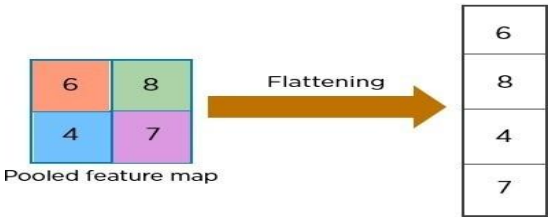


Figure 5.5 Flatten Layer

Flatening is the step in the process that follows the pooling layer. Flattening is a method of transforming all of the 2-Dimensional arrays as in figure 5.5 acquired from

pooled feature maps into a single long continuous linear vector, which is the output of the flattened layer.

The flattened matrix is sent into the fully connected layer, which classifies the picture accordingly.

5.1.6 Fully Connected Layer

The Fully Connected (FC) layer, which includes the weights and biases as well as the neurons, is used to link the neurons of the convolution neural network between two separate layers. All of these levels are typically placed before to the output layer and constitute the final few layers of a CNN Architecture. The preceding layers' input picture is flattened and supplied to the FC layer in the fully connected layer. The flattened vector is then passed across a few additional Fully linked layers, where the mathematical function's operations are normally performed. The categorization procedure begins during this step.

5.1.7 Softmax/Logistic Layer

The SoftMax function, which is also known as soft argmax or normalized exponential function is a generalization of the logistic function to classy multi-dimensions. This SoftMax function is used in multinomial logistic regression (like is the machine learning algorithm) and is often used as the last activation function of a neural network to normalize the output of a network to a probability distribution over predicted output classes, based on Luce's choice axiom

5.1.8 Output Layer

The output layer contains the label which is in the form of one-hot encoded which is the final output required

5.2. YOLO Description

YOLO [You Only Look Once: Unified, Real-Time Object Detection], a Deep Learning architecture suggested by Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi in the paper 'You Only Look Once: Unified, Real-Time Object Detection, takes a completely different approach. It's a sophisticated real-time convolutional neural network (CNN) for object detection. [15]

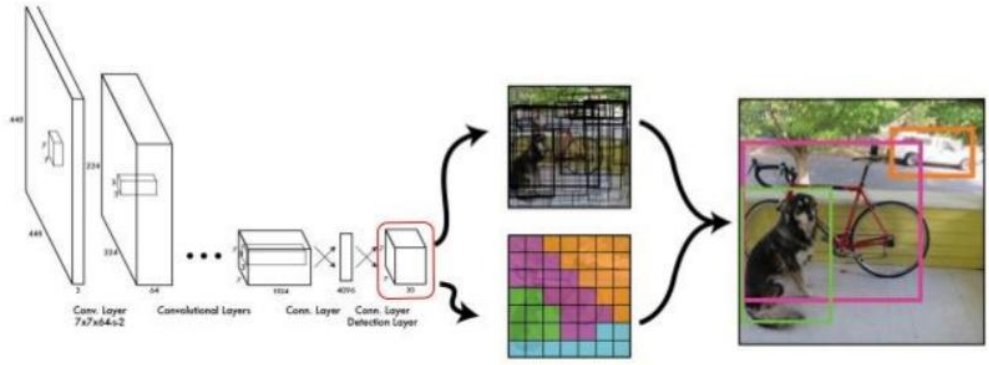


Figure 5.6 Classification using YOLO

It is especially popular because it has a high level of precision and can run in real-time or be utilised for real-time applications. To make predictions, the YOLO method "just looks once" at the input image, which means it only needs one forward propagation pass through the network.

5.2.1 How does YOLO works?

Localizers or classifiers are used by prior detection systems to carry out the detection procedure. The model is then used to apply to an image at various scales and places. For detections, only the portions of the image with a high score are examined.

The YOLO algorithm takes a different method. A single neural network is applied to the entire image using the method. The network then divides the image into areas, generating bounding boxes and predicting probabilities for each. The projected probabilities are used to weight the generated bounding boxes.

5.2.2 YOLO Architecture

Localizers or classifiers are used by prior detection systems to carry out the detection procedure. The model is then used to apply to an image at various scales and places. For detections, only the portions of the image with a high score are examined.

The YOLO algorithm takes a different method. A single neural network is applied to the entire image using the method. The network then divides the image into areas, generating bounding boxes and predicting probabilities for each. The projected probabilities are used to weight the generated bounding boxes.

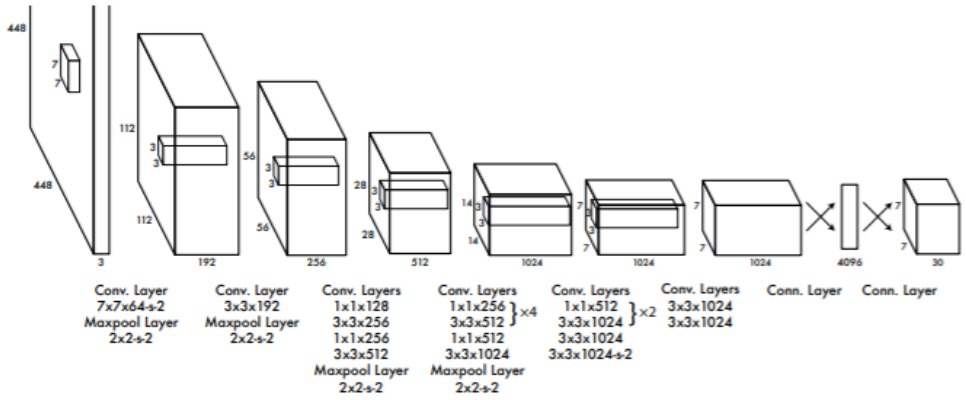


Figure 5.7 YOLO Architecture

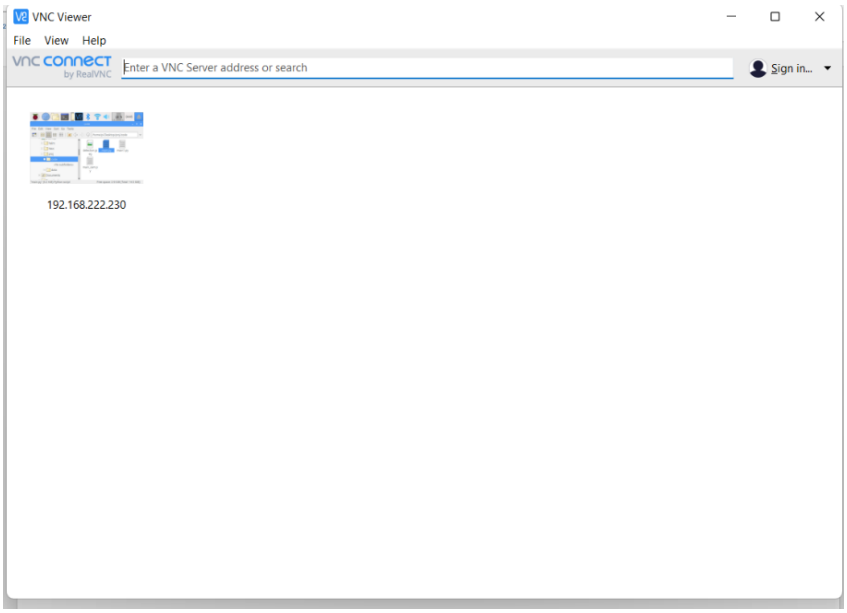
- The First 20 convolutional layers followed by an average pooling layer and a fully connected layer is pre-trained on the ImageNet dataset which is a 1000-class classification dataset.
- The pretraining for classification is performed on the dataset with the image resolution of 224 x 224×3.
- The layers comprise 3×3 convolutional layers and1x1 reduction layers.
- For object detection, in the end, the last 4 convolutional layers followed by 2 fully connected layers are added to train the network.
- Object detection requires more precise detail hence the resolution of the dataset is increased to 448 x 448
- Then the final layer predicts the class probabilities and bounding boxes.
- All the other convolutional layers use leaky ReLU activation whereas the final layer uses a linear activation.
- The input is of 448 x 448 image and the output is the class prediction of the detected object enclosed in the bounding box.

Next chapter will be on the execution procedure of the module.

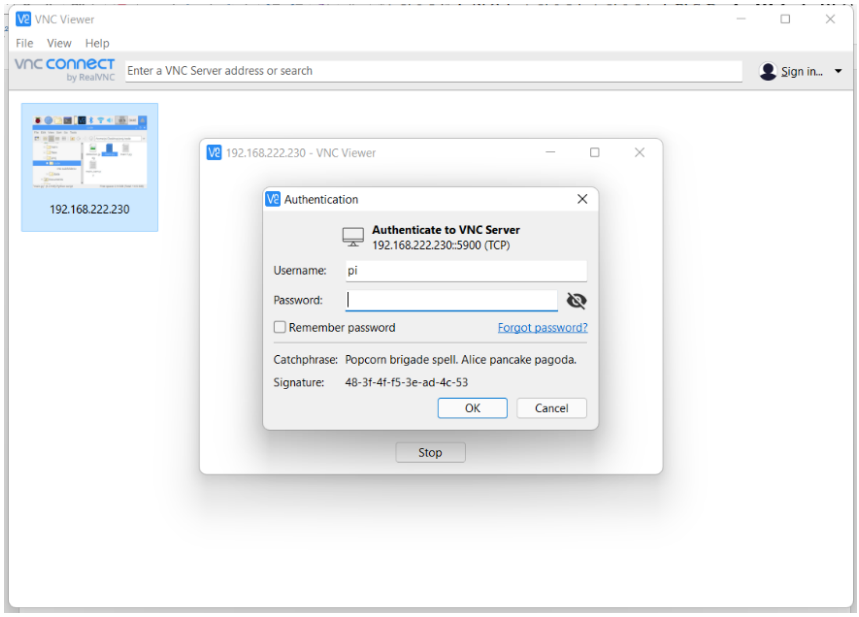
CHAPTER 6

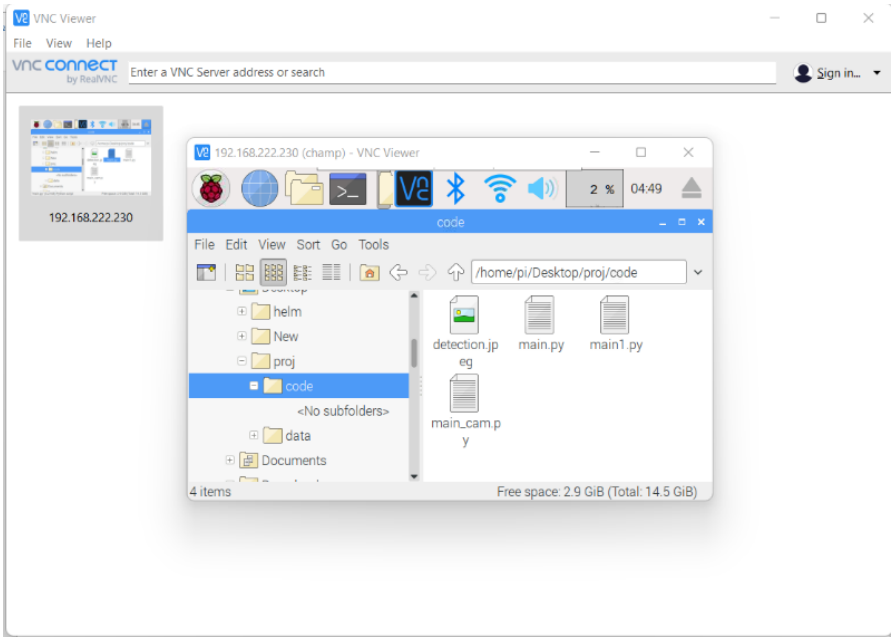
SOFTWARE EXECUTION PROCEDURE

a. To run the module first of all, connect the RPi with any of the supported platform. Here, a software named VNC Viewer is used.
It looks like below when launched

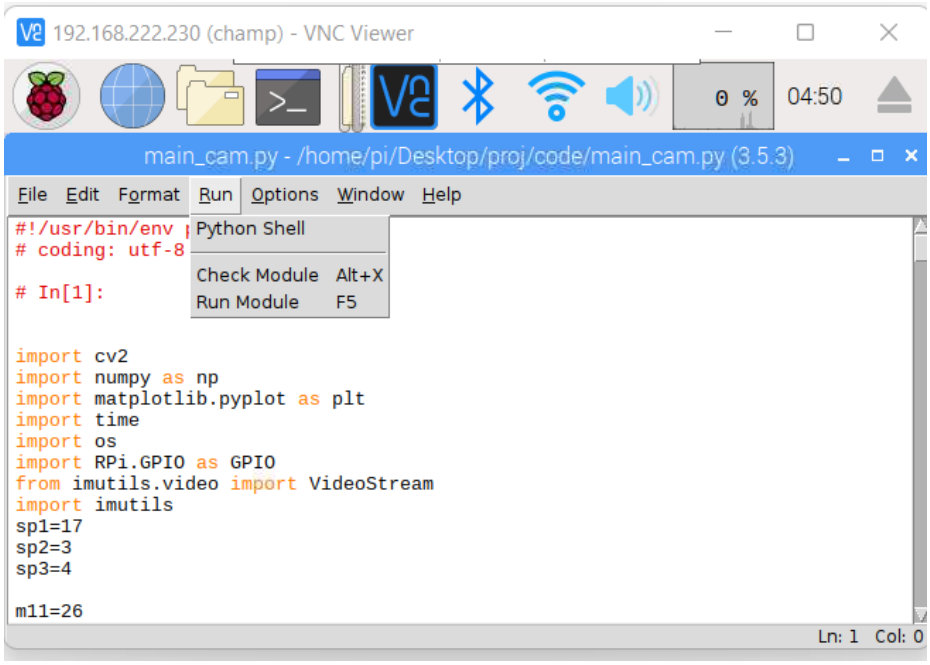


b. By entering the User Name and Password that was already set to the pi board one can access the desktop of the Pi board

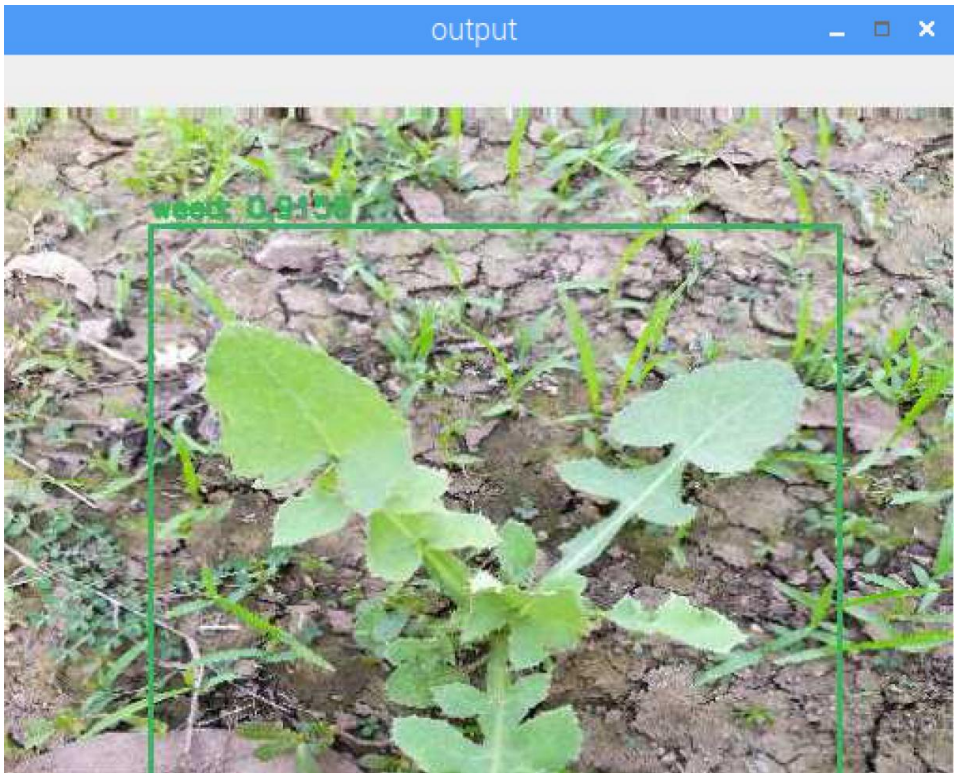




c. Open the python file and run the module

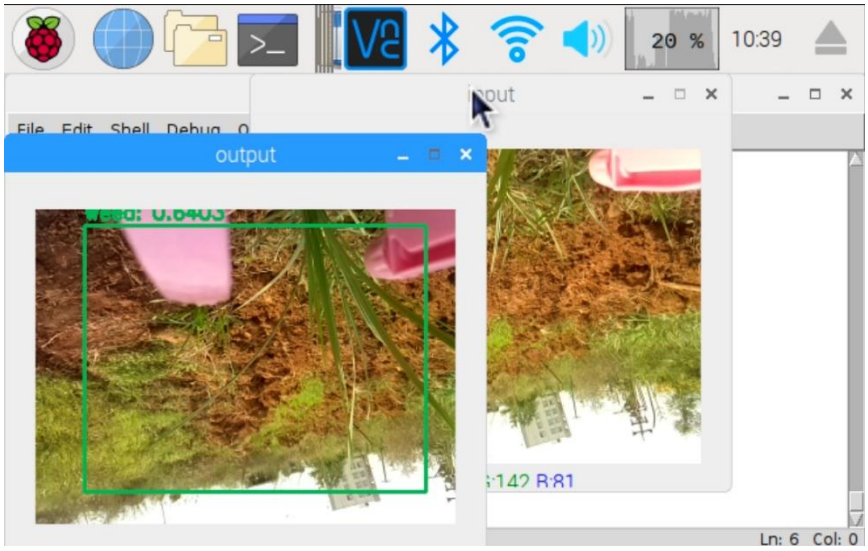


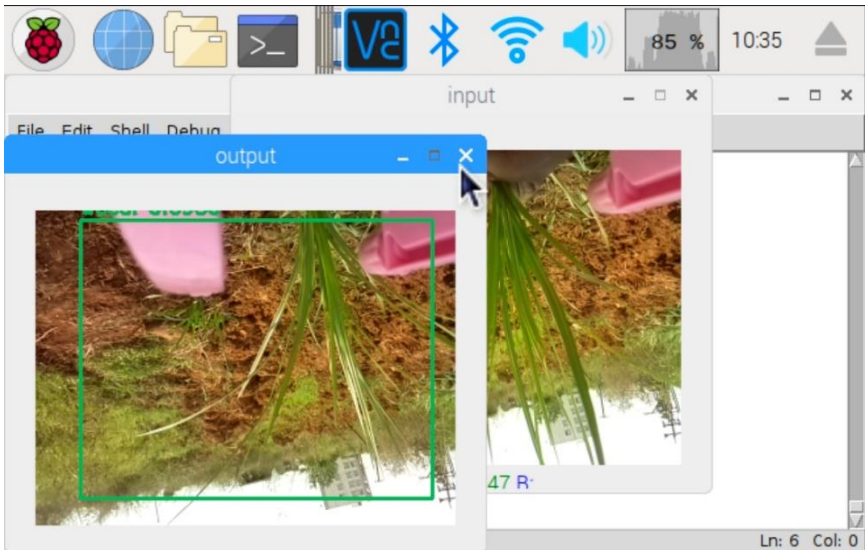
d. It will display the output



RESULTS

The described method for detecting weed species in organic farming using YOLO approach, detected the weeds in real-time. The detection speed can be adjusted to the applications accuracy requirement to maximize the detection speed. The proposed method shows that it is flexible and robust. If a crop leaf detected then the model moves to the next step and captures the next image. If no crop detected, then it considers it as weed and the sprayers will ON spray the herbicides on the weed.





Spaying on the spot where the weed was detected

Figure Spaying on the spot where the weed was detected

CONCLUSION AND FUTURE SCOPE

Weed detection is a crucial task in the agricultural productivity. This requires improved computational methods to provide a faster response. Thus, the method has higher accuracy compared to the existing methods. The experiment is carried out for sesame crop with multiple weeds. The result shows 75% accuracy in classification using convolutional neural network and maxpooling layers supported by reduced rate of misclassification of weed and crop.

The future work can be focused towards identification of weed species that can be combined with this existing work. With the LASER technology for instant killing (eradication) of the weed where ever it detected.

ANEXURE

```
#!/usr/bin/env python

# coding: utf-8

# In[1]:

import cv2

import numpy as np

import matplotlib.pyplot as plt

import time

import os

import RPi.GPIO as GPIO

from imutils.video import VideoStream

import imutils


sp1=17

sp2=3

sp3=4


m11=26

m12=19

m21=27

m22=13


# In[2]:
```

```
GPIO.setwarnings(False)

GPIO.setmode(GPIO.BCM) # Use BCM GPIO numbers


GPIO.setup(sp1,GPIO.OUT)

GPIO.setup(sp2,GPIO.OUT)

GPIO.setup(sp3,GPIO.OUT)


GPIO.setup(m11,GPIO.OUT)

GPIO.setup(m12,GPIO.OUT)

GPIO.setup(m21,GPIO.OUT)

GPIO.setup(m22,GPIO.OUT)


GPIO.output(sp1,True)

GPIO.output(sp2,True)

GPIO.output(sp3,True)


GPIO.output(m11,False)

GPIO.output(m12,False)

GPIO.output(m21,False)

GPIO.output(m22,False)


GPIO.output(m11,True)

GPIO.output(m12,False)
```

```
GPIO.output(m21,True)

GPIO.output(m22,False)


#load the class labels our YOLO model was trained on

labelsPath = '../data/names/obj.names'

LABELS = open(labelsPath).read().strip().split("\n")


# In[3]:

#load weights and cfg

weightsPath = '../data/weights/' + 'crop_weed_detection.weights'

configPath = '../data/cfg/crop_weed.cfg'


# In[4]:

#color selection for drawing bbox

np.random.seed(42)

COLORS = np.random.randint(0, 255, size=(len(LABELS), 3),dtype="uint8")


# In[5]:

print("[INFO] loading YOLO from disk...")

net = cv2.dnn.readNetFromDarknet(configPath, weightsPath)

vs = VideoStream(usePiCamera=True).start() # Raspberry Pi

time.sleep(3)
```

```
# In[6]:

i= 0


while True:

        #load our input image and grab its spatial dimensions

        #image = cv2.imread('../data/images/1.jpg')

        i=i+1

        image = vs.read()

        cv2.imshow('input',image)

        cv2.waitKey(1)

        print (i)

        x= 0

        w= 0

        out= 0

        if(i>300):

        i= 0


        GPIO.output(m11,False)

        GPIO.output(m12,False)

        GPIO.output(m21,False)

        GPIO.output(m22,False)


        time.sleep(2)
```

```
image = vs.read()
cv2.imshow('input',image)
cv2.waitKey(1)


(H, W) = image.shape[:2]
# In[7]:
#parameters
confi = 0.5
thresh = 0.5
# In[8]:
#determine only the *output* layer names that we need from YOLO
ln = net.getLayerNames()
ln = [ln[i[0] - 1] for i in net.getUnconnectedOutLayers()]
#construct a blob from the input image and then perform a forward
#pass of the YOLO object detector, giving us our bounding boxes and
#associated probabilities
blob = cv2.dnn.blobFromImage(image, 1 / 255.0, (512, 512),swapRB=True,
crop=False)
net.setInput(blob)
start = time.time()
layerOutputs = net.forward(ln)
end = time.time()
#show timing information on YOLO
```

```
print("[INFO] YOLO took {:.6f} seconds".format(end - start))
#initialize our lists of detected bounding boxes, confidences, and
#class IDs, respectively
boxes = []
confidences = []
classIDs = []
#loop over each of the layer outputs
for output in layerOutputs:
        #loop over each of the detections
        for detection in output:
                #extract the class ID and confidence (i.e., probability) of
                 #the current object detection
                 scores = detection[5:]
                 classID = np.argmax(scores)
                 confidence = scores[classID]
                 #filter out weak predictions by ensuring the detected
                 #probability is greater than the minimum probability
                 if confidence > confi:
                      #scale the bounding box coordinates back relative to the
                      #size of the image, keeping in mind that YOLO actually
                      #returns the center (x, y)-coordinates of the bounding
                      #box followed by the boxes' width and height
                      box = detection[0:4] * np.array([W, H, W, H])
```

```
                    (centerX, centerY, width, height) = box.astype("int")
                    #use the center (x, y)-coordinates to derive the top and
                    #and left corner of the bounding box
                    x = int(centerX - (width / 2))
                    y = int(centerY - (height / 2))
                    #update our list of bounding box coordinates, confidences,
                    #and class IDs
                    boxes.append([x, y, int(width), int(height)])
                    confidences.append(float(confidence))
                    classIDs.append(classID)
#apply non-maxima suppression to suppress weak, overlapping bounding
#boxes
idxs = cv2.dnn.NMSBoxes(boxes, confidences, confi, thresh)
#ensure at least one detection exists
if len(idxs) > 0:
        #loop over the indexes we are keeping
        for i in idxs.flatten():
                #extract the bounding box coordinates
                (x, y) = (boxes[i][0], boxes[i][1])
                (w, h) = (boxes[i][2], boxes[i][3])
                #draw a bounding box rectangle and label on the image
                color = [int(c) for c in COLORS[classIDs[i]]]
                cv2.rectangle(image, (x, y), (x + w, y + h), color, 2)
```

```
                out=LABELS[classIDs[i]]

                text = "{}: {:.4f}".format(LABELS[classIDs[i]], confidences[i])

                cv2.putText(image, text, (x, y - 5),
               cv2.FONT_HERSHEY_SIMPLEX,0.5, color, 2)

det = cv2.cvtColor(image,cv2.COLOR_BGR2RGB)

plt.figure(figsize=(12,8))

plt.imshow(det)

# In[9]:

#save detected image

path = 'detection.jpeg'#change if you want

det = cv2.cvtColor(det,cv2.COLOR_RGB2BGR)

cv2.imwrite(path,det)

print(x)

print(w)

print(out)

##if(out == 'weed'):

## if(x<w/3):

## GPIO.output(sp1,False)

## if((x>w/3) and x<(w/3)*2):

## GPIO.output(sp2,False)

## if(x>(w/3)*2):

## GPIO.output(sp3,False)

## time.sleep(5)
```

```
## GPIO.output(sp1,True)
## GPIO.output(sp2,True)
## GPIO.output(sp3,True)
if(out == 'weed'):
    GPIO.output(m11,False)
    GPIO.output(m12,False)
    GPIO.output(m21,False)
    GPIO.output(m22,False)

    if(x<50 and x != 0):
    GPIO.output(sp1,False)
    if((x>50) and x<100):
    GPIO.output(sp2,False)
    if(x>100):
    GPIO.output(sp3,False)
    time.sleep(5)
    GPIO.output(sp1,True)
    GPIO.output(sp2,True)
    GPIO.output(sp3,True)
    GPIO.output(m11,True)
    GPIO.output(m12,False)
    GPIO.output(m21,True)
    GPIO.output(m22,False)
```

```
if(out != 0):

cv2.imshow('output',det)

cv2.waitKey(1)

GPIO.output(m11,True)

GPIO.output(m12,False)

GPIO.output(m21,True)

GPIO.output(m22,False)

# In[ ]:
```

REFERENCES

[1] Umamaheswari S, Arjun R and Meganathan D. “Weed Detection In Farm Crops Using Parallel Image Processing”, Conference on Information and Communication Technology 2018.

[2] Maged Wafy, Egypt Hashem Ibrahim and Egypt Enas Kamel. “Identification Of Weed Seeds Species in Mixed Sample With Wheat Grains Using Sift Algorithm”, IEEE 2013.

[3] Xiaojun Jin, Jun Che, And Yong Chen. “Weed Identification Using Deep Learning And Image Processing In Vegetable Plantation”, IEEE 2020.

[4] C.Thirumarai Selvi, R.S.Sankara Subramanian and R. Ramachandran. ” Weed Detection In Agricultural Fields Using Deep Learning Process”, 7th International Conference on Advanced Computing & Communication Systems 2021.

[5] Siddhesh Badhan, Kimaya Desai, Manish Dsilva, Reena Sonkusare and Sneha Weakey. “Real-Time Weed Detection Using Machine Learning And Stereo-Vision”, 6th International Conference for Convergence in Technology 2021.

[6] Adnan Farooq, Jiankun Hu and Xiuping Jia. “Weed Classification In Hyperspectral Remote Sensing Images Via Deep Convolutional Neural Network”, IGARSS 2018.

[7] M.Dian Bah, Adel Hafiane, Raphael Canals and Bruno Emile. “Deep Features And One-Class Classification With Unsupervised Data For Weed Detection In Uav Images”, IEEE 2019

[8] Oscar Barrero, Diana Rojas and Christian Gonzalez. “Weed Detection In Rice Fields Using Aerial Images And Neural Networks”, IEEE 2016.

[9] Aichen Wang, Yifei Xu, Xinhua Wei and Bingbo Cui. “Semantic Segmentation Of Crop And Weed Using An Encoder-Decoder Network And Image Enhancement Method Under Uncontrolled Outdoor Illumination”, IEEE 2020.

[10] Inkyu Sa, Zetao Chen, Marija Popovic, Raghav Khanna, Frank Liebisch, Juan Nieto, and Roland Siegwart. “Dense Semantic Weed Classification Using Multispectral Images and MAV for Smart Farming”, IEEE Robotics and Automation Letters, Vol.3, No.1, January 2018.

[11] Hari Shankar RL, A.K. Veeraraghavan, Uvais, K.Sivaraman and S.Shreyas Ramachandran. “Application Of Uav for Pest, Weeds And Disease Detection Using Open Computer Vision”, International Conference on Smart Systems and Inventive Technology 2018.

[12] Vitali Czymmek, Leif O. Harders, Florian J. Knoll and Stephan Hussmann. “Vision-Based Deep Learning Approach For Real-Time Detection Of Weeds In Organic Farming”, IEEE 2019.

[13] https://opensource.com/resources/raspberry-pi

[14] https://www.ibm.com/cloud/learn/convolutional-neural-networks

[15] https://pjreddie.com/darknet/yolo/