City Research Online

City, University of London Institutional Repository

Citation: Abdulsalam, M. (2023). Artificial Intelligence based Robotic Platforms for Autonomous Precision Agriculture. (Unpublished Doctoral thesis, City, University of London)

This is the accepted version of the paper.

This version of the publication may differ from the final published version.

Permanent repository link: https://openaccess.city.ac.uk/id/eprint/31615/

Link to published version:

Copyright: City Research Online aims to make research outputs of City, University of London available to a wider audience. Copyright and Moral Rights remain with the author(s) and/or copyright holders. URLs from City Research Online may be freely distributed and linked to.

Reuse: Copies of full items can be used for personal research or study, educational, or not-for-profit purposes without prior permission or charge. Provided that the authors, title and full bibliographic details are credited, a hyperlink and/or URL is given for the original metadata page and the content is not changed in any way.

City Research Online: http://openaccess.city.ac.uk/ publications@city.ac.uk

Artificial Intelligence based Robotic Platforms for Autonomous Precision Agriculture

Mahmoud Abdulsalam

Supervisor: Prof. Nabil Aouf

School of Science and Technology

City, University of London

This dissertation is submitted for the degree of

Doctor of Philosophy
October 2023

I would like to dedicate this thesis to Allah the Almighty, my lovely parents, Dr. & Mrs. Abdulsalam, and my late aunty Ummu.

Declaration

I hereby declare that except where specific reference is made to the work of others, the contents of this dissertation are original and have not been submitted in whole or in part for consideration for any other degree or qualification in this, or any other university. This dissertation is my own work and contains nothing which is the outcome of work done in collaboration with others, except as specified in the text and Acknowledgements.

Mahmoud Abdulsalam
October 2023

Acknowledgements

First and foremost, I would like to acknowledge the efforts and support of my supervisor, Prof. Nabil Aouf. His commitment, discipline, expertise and motivation have been instrumental in achieving all the goals of this research. I have indeed learned a lot from him, and I am truly grateful for his guidance and the opportunities he has provided me.

I would also like to express my gratitude to Dr. Uthman for his unending support throughout this journey. His mentorship, encouragement, and emotional support have made this PhD journey easier. I am truly fortunate to know a person like him.

I would also like to appreciate all my colleagues and members of the RAMI group for their contributions, whether big or small. They have played a significant role in the completion of this research.

I extend my appreciation to my family members, Dr. Yidi, Mrs. Yidi, Halima, her children, my lovely sister Aisha, her baby, Muhammad Thani and Ibrahim, for their immense support of all kinds and their constant presence throughout my life. Their unwavering support has undeniably played a vital role in uplifting and motivating me to tackle this PhD.

I would also like to express my profound gratitude to my friends, especially Marzuq and Fatima, for their support and passion towards the success of this research. I appreciate them for being real and believing in me. I appreciate other friends and family not listed.

Finally, a special thanks to the Petroleum Technology Development Fund (PTDF) for the partial funding received for this research. It has been truly helpful and has ensured the completion of this PhD.

Abstract

Robotic applications are continuously expanding into every aspect of human livelihood, it becomes paramount to leverage this trend for precision agriculture. The agricultural sector despite being an important sector for human is slowly evolving in terms of technology. Crude and manual processes which are conventionally used for agriculture have severe economic and social impacts. The inefficiencies and less productiveness of these methods results to food wastage amidst food shortage, inconsistencies, time consumption, higher labour expenses, and low yield. The world will benefit from automating the processes in agriculture. In bid of addressing such, it becomes necessary to build on existing platforms and develop intelligent autonomous vehicles for precision agriculture. This should include development of intelligent drones for precision agriculture, development of intelligent ground robots for precision agriculture, and other systems working cooperatively. To achieve this, we leverage on Artificial Intelligence (AI) and mathematical methods to impact sufficient intelligence on robotic platforms to make them suitable for precision agriculture.

This thesis explores the capabilities of AI for weed classification and detection, weed relative position estimation, fruit 6D pose estimation and virtual reality for teleoperated systems in fruit picking. Infestation of weeds diminishes the yield of crops in agriculture. Deep learning is becoming a more popular approach for identifying weeds on farmlands. However, precision agriculture requires that the object of interest (weed) is precisely classified and detected to facilitate removal or spraying. An approach for this is presented and involves cascading a classification network (ResNet-50) with a detection network (YOLO) for weed classification and detection which we termed Fused-YOLO. Thus, weeds can precisely be located and classified (type) within an image frame.

Inspired by the precision of this detection model, the work extends to presenting a novel monocular vision-based approach for drones to detect multiple types of weeds and estimate their positions autonomously for precision agriculture applications. A drone is subjected to an elliptical trajectory while acquiring images from an onboard monecular camera. The images are fed to the fused-YOLO model in real-time. The centre of the detection bounding boxes is leveraged to be the centre of the detected object of interest (weeds). The centre pixels are extracted and converted into world coordinates forming azimuth and elevation angles from the target to the UAV and are effectively used in an estimation scheme that adopts the Unscented Kalman Filteration to estimate the exact relative positions of the weeds. The robustness of this algorithm allows for both indoor and outdoor implementation while achieving a competitive result with affordable off-the-shelf sensors.

Artificial intelligence for autonomous 6D pose estimation has valuable contributions to agricultural practices rallying around fruit picking, harvesting, remote operations and other contact-related applications. Conventionally, Convolutional Neural Networks (CNNs) based approaches are adopted for pose estimation. However, precision agriculture applications are demanding on higher accuracy at lower computational costs for real-time applications. Motivated by this, a novel architecture called Transpose is proposed based on transformers. TransPose is an improved Transformer-based 6D pose estimation with a depth refinement. More modalities often result in higher accuracy at the expense of computational cost. Transpose takes in a single RGB image as input without extra modality. However, an innovative light-weight depth estimation network architecture is incorporated into the model to estimate depth from an RGB image using a feature pyramid with an up-sampling method. A transformer model having proven to be efficient, regress the 6D pose directly and also outputs object patches. The depth and the patches are utilised to further refine the regressed 6D pose. The performance of the model is extensively assessed and compared with state-of-the-art methods. As part of this research, a first-ever fruit-oriented 6D pose dataset was acquired.

Lastly, a seamless teleoperation pipeline that interfaces virtual reality with robots for precision agriculture tasks is proposed to pave the way for virtual agriculture. This utilises the Transpose model to estimate the 6D pose of a fruit and render it in a virtual reality environment. A robotic manipulator is which is then controlled from within the virtual reality environment to pick/harvest the fruit while being guided by the Transpose AI model. The robustness of the pipeline is tested over simulation and real-time implementation with a physical robotic manipulator is also investigated.

Table of contents

List of figures

List of tables

Acronyms

ADS Average Detection Score.

AI Artificial Intelligence.

ANN Artificial Neural Network.

AP Average Precision.

AR Augmented Reality.

CAD Computer Aided Design.

CNN Convolutional Neural Network.

CPU Central Processing Unit.

DEN Depth Estimation Network.

DETR Detection Transformer.

DInSAR Differential Synthetic Aperture Radar interferometry.

DNN Deep Neural Network.

DRT Detection and Regression Transformer.

ESC Electronic Speed Control.

Fast R-CNN Fast Region-based Convolutional Neural Network.

Faster R-CNN Faster Region-based Convolutional Neural Network.

FC Fully Connected.

FFN Feed Forward Network.

FOV Field Of View.

FPN Feature Pyramid Network.

FPS Frames per Second.

GIS Geographic Information Systems.

GPS Global Positioning System.

GPU Graphics Processing Unit.

GUI Graphical User Interface.

ILSVRC ImageNet Large Scale Visual Recognition Challenge.

IMU Inertial Measurement Unit.

IOU Intersection Over Union.

LAI Leaf Area Index.

LiDAR Light Detection and Ranging.

ML Machine Learning.

MLP Multilayer Perceptron.

MSE Mean Squared Error.

NDVI Normalized Difference Vegetation Index.

NIR	Near Infrared.
ReLU	Rectified Linear Unit.
ROI	Region of Interest.
ROS	Robotics Operating System.
RPN	Region Proposal Network.
SGD	Stochastic Gradient Descent.
SIFT	Scale-Invariant Feature Transform.
SURF	Speeded Up Robust Features.
UAV	Unmanned Aerial Vehicle.
UGV	Unmanned Groud Vehicle.
UKF	Unscented Kalman Filter.
VGG	Visual Geometry Group.
VR	Virtual Reality.
YOLO	You Only Look Once.

Chapter 1

Introduction

Chapter abstract

This chapter introduces the reader to the research context that forms the inspiration behind this thesis. It provides a background knowledge of precision agriculture which includes the evolution of agricultural practices so far and the need for intelligent vehicles in precision agriculture. This chapter gives an overview of the overall research, the scope, and the research aim and objectives. Artificial intelligence as a tool for intelligent robotics platforms is discussed while analysing the various applications of such tool in the agriculture domain.

1.1 Motivation

The demand for food keeps getting higher by the day as the global population increases while the availability diminishes. One in four people globally is moderately or severely food insecure (Max and Hannah, 2013). Productivity is lost as a result of pests, weeds, disease infestation, cost of labour, inefficiencies in processes, etc. Crop losses due to pests, weeds and diseases are a major threat to the incomes of rural families and to food security worldwide (Savary and Willocquet, 2014). Statistics show that pests, weeds and diseases can lead to high primary yield losses (26%) and even higher secondary yield losses (38%) (Cerda et al., 2017).

Another challenge that bothers agriculture is the shortage of labour. Nowadays, the shortage of labour for agriculture is growing as many people tend to move to white-collar jobs. As a result, labour becomes scarce and expensive to seek. An alternative opted for by most farmers involves the use of farm machinery. However, most farm machinery being heavy types of equipment are powered by fossil fuels thereby releasing much carbon footprint to the atmosphere. These carbons deplete the ozone layer and add to global warming through greenhouse gases emission. They also come with an expensive price tag.

Pertinent to the listed problems, this research explores the gateway to intelligent vehicles in a bid to increase crop yield while decreasing the labour deficit. Robotic platforms are becoming more popular for military, medical, and space applications. This can be due to their ability to perform efficiently and emit zero carbons into the atmosphere. Although robotics in agriculture is still maturing, they can help address labour and productivity challenges via precision agriculture. Unlike other robotics applications, most of agricultural activities are complex and diverse in nature. Moreover, an optimal standalone platform is one which can perform multiple agricultural activities. This can be very challenging due to the diverse nature and different intricacies associated with each task. Thus, having a robotic platform alone is insufficient to address agriculture's challenges. An intelligent vehicle that is able to classify, detect, and make decisions like humans can be explored as the solution to those challenges. Complex farming processes like weed identification and removal, precise harvesting etc require more than just a robotic platform. Artificial intelligence simulates human-like intelligence in terms of learning and thinking. It involves learning patterns and analysing datasets to implement tasks that normally involve human intelligence. Provided these robotic platforms are intelligent, the activities that involves human intervention such as classification, detection, and localisation in precision agriculture can be tackled efficiently thereby increasing productivity and food security.

1.2 Precision Agriculture

Agriculture can be defined as the science and art of cultivating the soil, including the allied pursuits of gathering crops and rearing livestock; tillage, husbandry, and farming (in the widest sense) (Harris and Fuller, 2014). An agricultural production system is the outcome of a complex interaction of seed, water and agrochemicals including fertilizers and pesticides. Therefore, careful management of all inputs is essential for the sustainability of such a complex system (Dwivedi et al., 2017). Environmental degradation is more eminent these days as the productivity of agriculture is enhanced without taking cognisance of the cost to the environment. Common examples of agricultural-caused degradation include soil degradation, tillage erosion, more dangerously, agriculture plastic waste and agrochemicals. The data from (Mace et al., 2017) summarized in Fig. 1.1 suggests that at least 650 tonnes of pesticides are used on crops yearly in the United Kingdom alone. 56% of these pesticides are used as herbicides to treat weeds in the farms as in Fig. 1.2. Varying from general weeds, broad-leaf weeds, town weeds, etc. A lot of volume of these environmentally harmful chemicals are used to treat these weeds. As suggested by the data from (Mace et al., 2017), an average of 450,000 hectares are covered in herbicides every year (see Fig. 1.3).

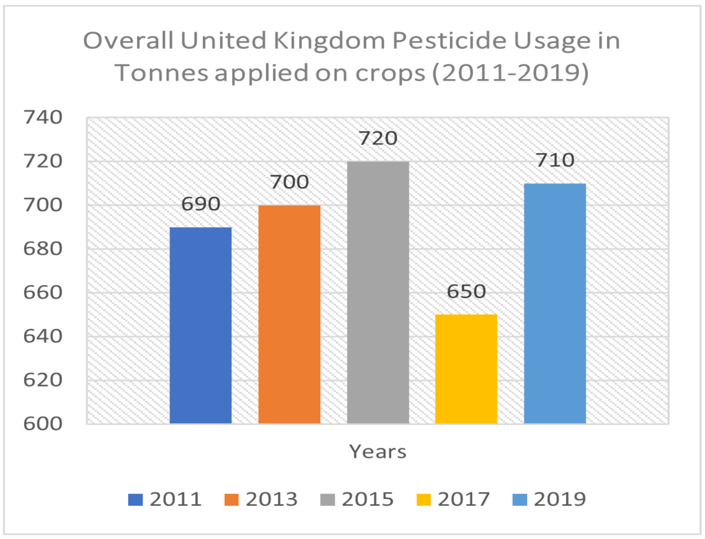


Fig. 1.1 Pesticides usage on crops in the United Kingdom.

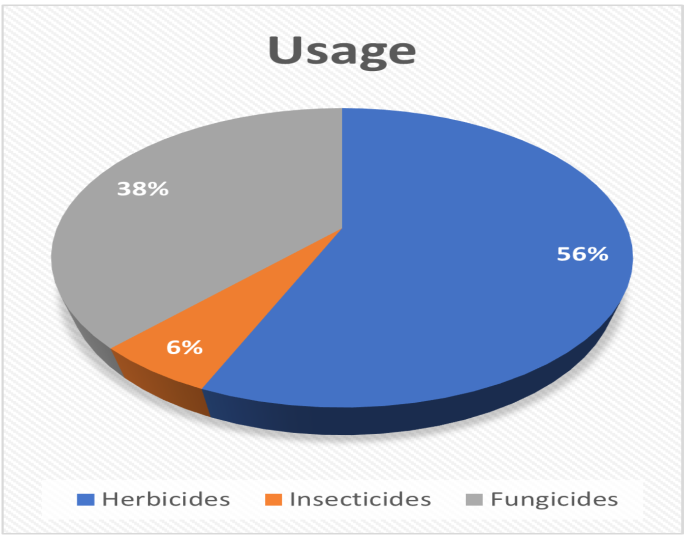


Fig. 1.2 Pesticides usage on crops according to type.

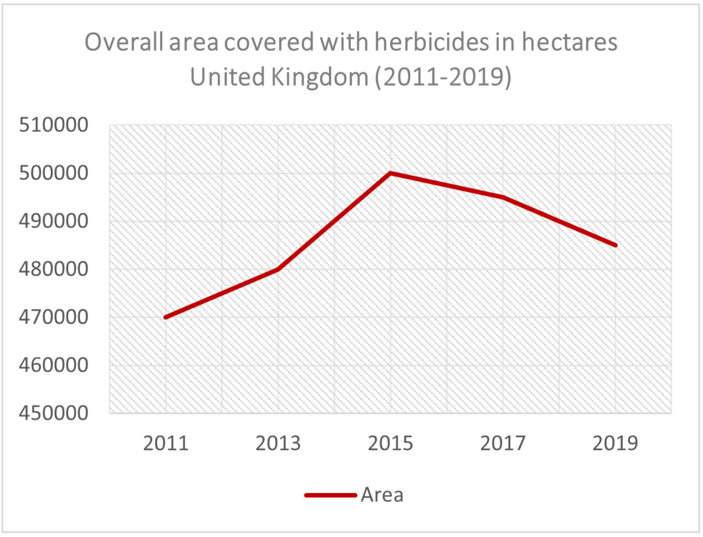


Fig. 1.3 Herbicide area coverage for weed treatment

To manage the excessive use of agrochemicals in the environment and other inefficient practices, precision agriculture is introduced. The success of precision agriculture depends on the accurate assessment of the variability, its management, and evaluation in the space-time continuum in crop production (Dwivedi et al., 2017).

Definition 1 ***Precision Agriculture***

Precision Agriculture is the application of technologies and principles to manage spatial and temporal variability associated with all aspects of agricultural production for improving production and environmental quality.

Precision agriculture involves the adequate and optimum usage of resources based on various parameters governing crop yield (Balaji et al., 2018). Precision agriculture has an edge over conventional practices due to some of these advantages:

- Better decision making due to availability of record
- Minimize farm degradation
- Increase productivity and food security
- Reduced cost of production
- Improved Farm Management
- Scalability and Automation

As much as precision agriculture is important, executing it effectively is more important. Although agriculture is a vast field, the scope can be narrowed to meaningful research which is using artificial intelligent vehicles for autonomous vehicles in precision agriculture. In a bid to elevate the agricultural sector and fit into the modern technological trend, there is a need to address certain problems that are labour-intensive and costly. These activities are generally done manually and hence allowing so much room for human error. It becomes necessary to address such problems through precision agriculture. In fact, some countries have already subscribed to this practice and are already experiencing substantial success in production. The data provided by (Rajmane et al., 2020) which shows land distribution and production is plotted in Fig. 1.4 and 1.5. It proves that developed countries that are already subscribing to early precision agriculture are able to achieve more productivity than under-developed and developing ones.

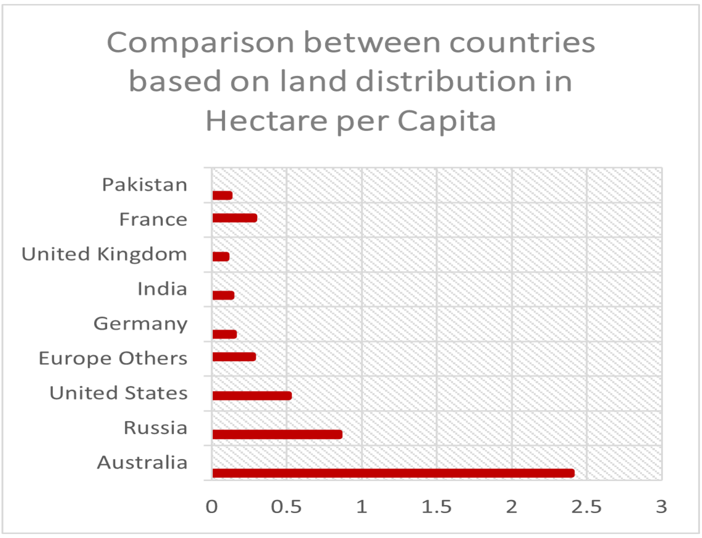


Fig. 1.4 Comparison between countries based on land distribution

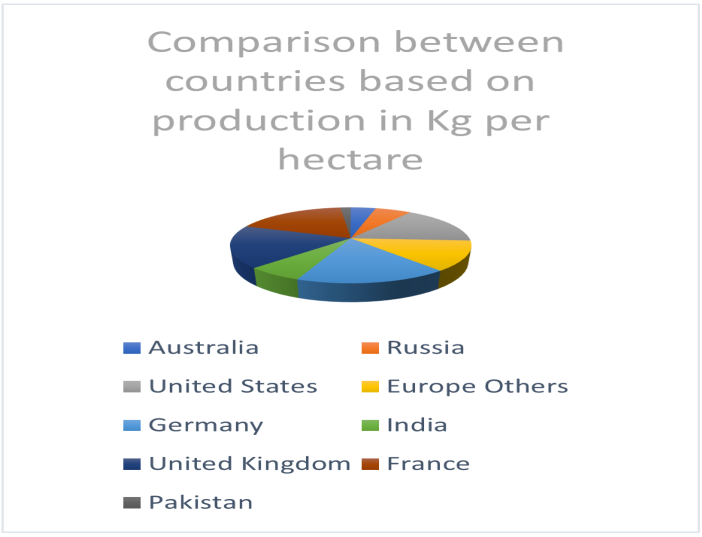


Fig. 1.5 Comparison between countries based on production

From this data, it is clear that developed countries such as the United States and Germany lead production capability despite having fewer land distribution. This is due to the proper implementation of precision agriculture. Whereas countries like Pakistan and India are lagging mostly because of the conventional approach to agriculture. This analysis suggests that even though the breakthrough in precision agriculture is not yet state-of-the-art, it is already providing solutions with minimal inputs. The main aim of precision agriculture is to increase agricultural profits with minimum inputs, reducing the wastage of fertilizers, herbicides and manpower (Rajmane et al., 2020). Robotics on the other hand aims at performing tasks with the maximum possible efficiency using sensors. The introduction of robotics to agriculture makes a perfect combination for precision agriculture. In this method, advanced and modern technologies such as computer vision using cameras, Global Positioning System (GPS), Artificial Intelligence (AI), remote sensing and most importantly robotics are introduced to make farming more productive and precise. Robotics can easily be adopted in precision agriculture in areas like site selection, soil preparation, field preparation, seeding and planting, watering, fertilizer and pesticide application, weed removal and harvesting as suggested by (Rajmane et al., 2020). Table 1.1 compares between conventional agriculture and precision agriculture on common agricultural tasks.

1.3 Robots for Precision Agriculture

As forecasted by (Mahsa, 2020), the global market for agricultural robots is currently on the rise as shown in Fig. 1.6. It becomes pertinent to explore this technological trend and improve on existing solutions. Agricultural robots mainly comprise drones, Unmanned Ground Vehicle (UGV)s and manipulators. Each can be used separately or cooperatively to achieve maximum optimization and efficiency.

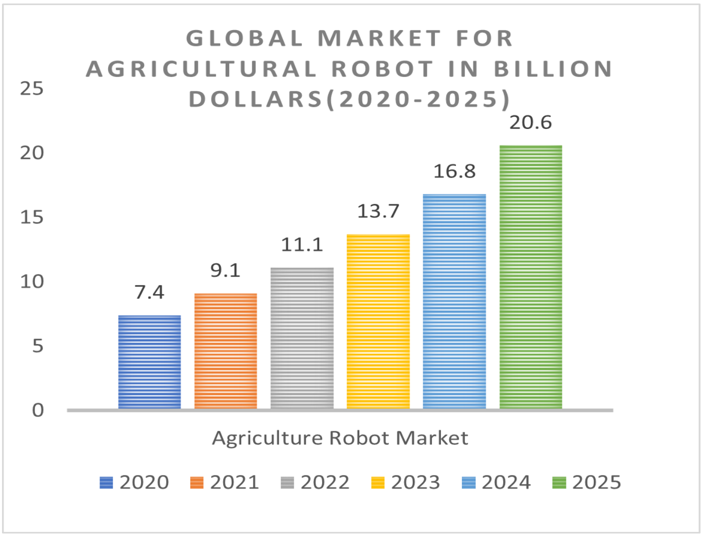


Fig. 1.6 Forcasted Market for Agricultural robots showing a continuous upward trend

Table 1.1 Comparison between Conventional and Precision Agricultural Practices.

Conventional Agriculture	Precision Agriculture
Site Selection	
Done manually	Done with drones and GPS systems
Soil Preparation	
Done based on trial and error, previous experience and expert hiring	Done with sensors like temperature sensor, Humidity Sensor, Soil Moisture Sensor etc
Field Preparation (Planking and Ploughing)	
Done with bullocks and tractors	Done using Agricultural robot for automatic ploughing
Seeding and planting	
Done Manually with Hand tools	Done using Precision drills, Broadcast seeders, Seed drills, Air seeders, Robotic planters
Watering	
Done with Drip Irrigation systems or manually	Done with Drip Irrigation using Internet of Things
Fertilizer and Pesticide application	
Done with Hand spray and manually	UAVs and UAVs, Sprayers, GPS, Smartphone Applications, and Remote sensing is used
Weed Removal	
Using hand tools	Advanced robots for weeding are used
Harvesting	
Manual picking	Robotic pick and place arm, Mechanical harvesting, limb shaker, canopy shaker, Abscission Chemical

Drones are important when it comes to monitoring, classification and detection. Their ability to rise above ground level and have an aerial view of a field makes them the best choice for several monitoring purposes by equipping them with the appropriate sensors for the task. (Oré et al., 2020) work on Crop Growth Monitoring focuses on Accurate, high-resolution maps for crop growth monitoring needed by precision agriculture. They utilized differential synthetic aperture radar interferometry (DInSAR) and a drone to monitor the growth of coffee, sugarcane and cone. This work obtained growth deficit maps with an accuracy down to $5cm$ and a spatial resolution of $1m$. The work proves that agriculture can greatly benefit from drones in applications like crop growth monitoring, detection and by extension used for other agricultural tasks involving aerial vision. However, the complexity and resource intensiveness of this approach makes it expensive to use. (Balaji et al., 2018) proposed a multifunctional drone using cheap sensors by designing an Unmanned Aerial Vehicle (UAV) for Crop, Weather Monitoring and Spraying Fertilizers and Pesticides. The work utilised a hexacopter with Raspberry Pi, a water level sensor, a temperature and humidity sensor, and a sprinkler. No evaluation of this system was made with regard to real-time implementation. The results of this work were not published and hence do not give a basis for evaluation. However, the work shows the possibility of using cheap drone setups and systems to achieve autonomous farming in contrast to (Oré et al., 2020). (Thapa, 2021) gave an overview of UAV applications and their potential for making agriculture smart, their current status, and the scenario for application. The review highlighted that drone hardware implementations are purely dependent on critical aspects like weight, range of flight, payload, configuration, and their costs, research technologies, methods, systems, and limitations of the UAVs. This gets to show that the type of drones used in agriculture solemnly depends on the tasks to be carried out. This calls for an in-depth review on the types of drones currently in development. (Puri et al., 2017) discussed the various types of UAV for precision farming in their work entitled *Agricultural drones: a modern breakthrough in precision agriculture*. They highlighted the importance of drones in agriculture for farm analysis, time-saving, higher agricultural yield, Geographic Information Systems (GIS) mapping integration, and imaging of crop health status. The agricultural drone available in the market discussed in their work is summarized in table 1.2 below.

Table 1.2 Agricultural Drones and their descriptions.

Drones	Description
Honeycomb AgDrone System	The AgDrone System is developed by the Honeycomb Company and is regarded as the most sophisticated drone for Agriculture. It can cover up to 600-800 acres of field every hour while flying at 400 feet.
DJI Matrice 100	DJI Matrice 100 is regarded as the Best Quadcopter based Drone for Agriculture with dual battery support, which increases almost 40 minutes of flight times. This drone has capabilities like GPS, Flight Controller, DJI Lightbridge which is regarded as Advanced Flight Navigation System to do complex tasks and is easy to operate in all environmental conditions.
DJI T600 Inspire 1	The DJI T600 Inspire Quadcopter is another powerful Agriculture drone known for its fast charging. It specially features 4K Video recording, individual flight and camera control and provides easy navigational capabilities.
Agras MG-1- DJI	Agras MG-1 – DJI is the ultimate octocopter designed for assisting farmers in spraying large areas of farmland with pesticides, insecticides or Fertilizers. The unique feature of this drone is that MG-1 is compatible to carry up to 10KG of liquid payloads and can cover 4000-$6000m^2$ area in just 10 minutes which is regarded as 70 times faster as compared to manual spraying. MG-1 has a fully sealed body and consists of an efficient, integrated centrifugal cooling system to keep the air flowing to each part of the Drone during flight time. MG-1 is equipped with 4 nozzles for accurate spraying of fertilizer in the field.
EBEE SQ- SenseFly	The EBEE SQ is a High-Performance agriculture drone designed for Crop Monitoring from planting to harvest to assist farmers in better crop yield. This drone is fully integrated and highly precise and has a multispectral sensor capable of capturing data across four non-visible bands together with RGB imagery in just a single flight. It provides a larger coverage as compared to other quadcopter drones and has automatic 3D flight planning. It has compatibility with Pix4dmapper AG mapping software to create Vegetation Index maps for crop fields and identify problem areas during flight.

Although some of these drones are highly efficient for their specific type of tasks, they are not capable of generalising to other tasks. Moreover, most of these drones are operated manually. Automation can be achieved by allowing the robot to *"think"*. This is possible through AI. Now the question remains *"What type of drone is suitable for agriculture?"*. The data from (DataMIntelligence, 2020) suggests that most of the drones used in agricultural practices are multi-rotor drones then followed by Fixed wing drone types. This can be seen statistically in Fig. 1.7.

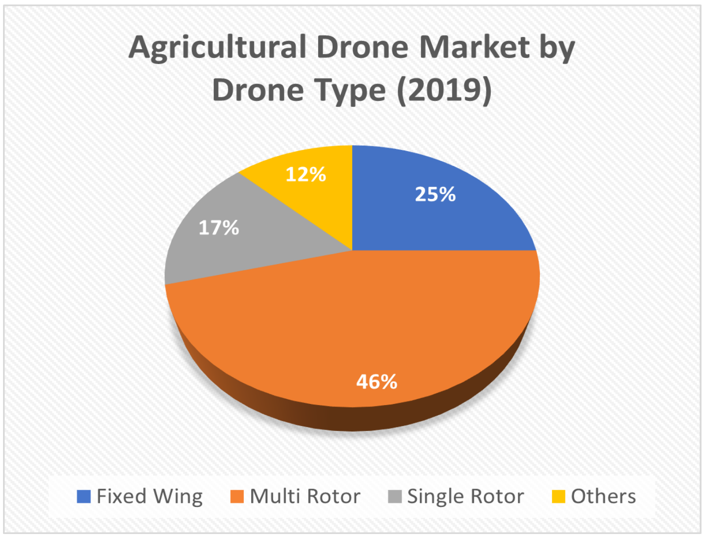


Fig. 1.7 Agricultural Drone Market by types

The basic necessities of a drone for precision agriculture include a frame, brushless motors, Electronic Speed Control (ESC) modules, a control board, an Inertial Navigation System (ESC), and a transmitter and receiver. Drones cannot function without sources of information to communicate with the environment, these are sensors. As iterated by (Daponte et al., 2019) in the case of precision agriculture, the sensors embedded in drones include multispectral cameras, thermal cameras, RGB cameras and Light Detection and Ranging (LiDAR) systems. Multispectral cameras are used for quantifying the state of the monitored vegetation in terms of chlorophyll content, leaf water content, ground cover and Leaf Area Index(LAI), and the Normalized Difference Vegetation Index (NDVI). Thermal cameras have demonstrated a high potential for the detection of water stress in crops due to the increased temperature of the stressed vegetation. LiDAR can be used to map the farm area and efficiently navigate robotic platforms in the farm. From the literature, it is fair to conclude that most of the drones utilised for agriculture are multi-rotor. Thus, this work utilises an affordable Parrot ARDrone and a custom-made drone equipped with the necessary sensors to be discussed in the subsequent section.

The growing population demands a more sustainable agriculture practice. Following the successes of ground robots for military and disaster applications, various research has emerged on how to adopt such vehicles for agriculture practices. (Mueller-Sim et al., 2017) in their work suggested The Robotanist: A Ground-Based Agricultural Robot for High-Throughput Crop Phenotyping. This robot is capable of measuring the physiological and morphological traits (phenotypes) of crops in outdoor test plots. These processes are often labour-intensive and have too many inconsistencies. (Mueller-Sim et al., 2017) presented a novel ground robot that is capable of navigating through row crops such as maize autonomously. The robot is capable of measuring phenotypic information with the aid of a modular array of non-contact sensors. The strength of the stalks is measured by the robot by deploying a manipulator. (Grimstad et al., 2015a) developed a multipurpose ground robotic platform. This platform is capable of performing activities like seeding, harvesting and weeding. Their work entitled “the design of a low-cost, lightweight, and highly versatile agricultural robot” is economically viable, has lightweight so as to operate on soil without damaging the structure of the soil. It has some essential tools attached to it including seeding tools, liming, weeding tools, and crop scouting tools. Ground robots can perform energy-demanding tasks as portrayed by (Grimstad et al., 2015b). Tasks performed by heavy machinery such as tractors can also be performed with light ground robots. This robot named *Thorvald* is capable of performing common agricultural practices such as seeding and crop monitoring. This platform was used by researchers working on cereal crop phenotyping and the results obtained were generally fair. However, it could not match up with the expectations of the researchers. (R Shamshiri et al., 2018) conducted research on the developments in agricultural robotics. The work iterated the importance of robotics in agriculture and highlighted some robots with their description summarised in table 1.3.

Table 1.3 Agricultural Ground Robots and their Descriptions

Ground Robots	Description
SWEEPER	The system is made up of a collection of independent mobile platforms equipped with a robot manipulator that holds an end-effector and a fruit-grabbing device for harvesting.
BoniRob	It is an integrated multipurpose farming robotic platform for row crops weed control developed by interdisciplinary teams. It is also capable of creating a detailed map of the field.
AgBot II	It is an innovative ground robot prototype developed by the Queensland University of Technology for several purposes such as autonomous fertilizer application, weed detection and classification, and mechanical or chemical weed control.
Autonome Roboter	a research effort robot developed by Osnabrück University of Applied Sciences for weed control.
Tertill	It is powered by solar and fully autonomous. It is a compact robot developed by Franklin Robotics for weed cutting.
Hortibot	a robot developed by the University of Aarhus for transporting and attaching a variety of weed detection and control tools such as cameras, herbicide and spraying booms
Kongskilde Robotti	a robotic platform equipped with a drive belt operating based on the FroboMind software that can be connected to different modules and can be implemented for automated and semi-automated mechanical weed control, precision seeding, furrow opening and cleanings.
RIPPA	Developed by the Australian Centre for field robotics at Sydney University. It is a solar-powered Robot for Intelligent Perception and Precision Applications.
Ladybird	an autonomous multipurpose farm robot for surveillance, mapping, classification and detection of different vegetables.
Wall-Ye	a prototype vineyard robot for pruning, mapping, and possibly harvesting the grapes
MARS	They are mobile agricultural robot swarms. Small and streamlined mobile robot units that have minimum soil compaction and energy consumption and aim at optimizing plant-specific precision agriculture
Vine agent	developed by the Universitat Politècnica de València. It is equipped with advanced sensors and artificial intelligence to monitor the field for plant health assessment.
HV-100 Nursery Bot	A light weighted robot developed by Harvest Automation for moving plants and potted trees in greenhouses and small orchards.
GRAPE	Ground Robot for vineyard monitoring and Protection. It can perform smart autonomous navigation, plant detection and health monitoring, and manipulation of small objects.

Some complex tasks will require the fusion of both platforms (UAVs and UGVs) to successfully implement the agricultural tasks intended. This approach becomes even more useful when robots begin learning from each other and improve their performance over time (R Shamshiri et al., 2018). In order to explore multi-platform cooperation, (Pretto et al., 2020) proposed building an aerial-ground robotics system for precision farming. This setup was designed to monitor crop density, weed pressure, and crop nitrogen nutrition status, and to accurately classify and locate weeds. The researchers further introduced navigation and mapping systems to address the specificity of the employed robots and the agricultural environment, highlighting the collaborative modules enabling the UAVs and UGVs to collect and share information in a unified environment model. The test was performed on different crops and different fields. The concept was to use the UAV to monitor the farm while communicating with the ground robot to perform selective spraying autonomously. Still on the cooperation, (Potena et al., 2019) introduced the *AgriColMap: an aerial-ground collaborative 3D-Mapping for precision agriculture*. *AgriColMap* is a novel map registration pipeline that leverages a grid-based multi-modal environment representation, including a vegetation index map and a Digital Surface Model. Maps built from both the aerial and ground robot are collected, and the dominant coherent flows are selected using a voting scheme and are used as point-to-point correspondences to infer a preliminary non-rigid alignment between the maps. A final refinement is then performed by exploiting only meaningful parts of the registered maps. (Tokekar et al., 2016) proposed a sensor planning for a symbiotic UAV and UGV system for precision agriculture which also involves the collaboration of both UAVs and UGVs. The idea was to use the UGV to deploy the UAV at carefully selected locations. As the UAV is taking images, the UGV will take soil measurements nearby. The UAV can then land on the UGV surface which will then take the UAV to the next deployment location. Landing and ascending consume energy, leading to the problem of choosing how often to land. To tackle this, they developed an algorithm that optimally decides how frequently the UAV lands.

Summarily, All these field robots show that research is beginning to penetrate the agriculture sector. However, most of the robots mentioned are at the research stage and not yet deployable for large-scale commercial purposes while others are not affordable. Even though the automation of agriculture is inevitable as food demand grows, there is still a long way to go in making such platforms ready for market at an affordable price. Thus, utilising affordable platforms with basic sensors yet equipping them with the required level of intelligence can perhaps narrow the gap of automation for agriculture as targeted by this thesis.

1.4 Research Objectives

Given the established background and the challenges listed in this introductory chapter, the objective of this thesis can now be defined. The aim is to investigate the fusion of artificial intelligence and robotic platforms so they can carry out farming operations autonomously. Artificial intelligence techniques such as machine learning and computer vision are essential in this journey. Fetching data from several sensors and processing via AI in a befitting way will give rise to an intelligent system that can think and execute actions in a human-like manner. Although farming operations are vast and dynamic in nature, their similarities are tied to the techniques used in implementing them. After thorough investigation, this thesis identifies three key elements that are common in the majority of the agricultural activities. These elements are believed to be the backbone of the majority of agricultural activities. In essence, making these elements intelligent will automatically make the agricultural activities intelligent. These elements are: classification, detection and pose estimation. A simple case study of perception-based inference can be that of weed removal. Weeds often result in low yields in crops. The two major processes of weed control are identification (classification and detection) and removal based on relative pose estimation. Manual processes are often less efficient, labour intensive, and too much time wastage. However, adopting AI for this task includes the following steps:

- Step 1: Detect the weed (AI for detection)
- Step 2: Classify the type of weed (AI for classification)

- Step 3: Estimate the relative pose for removal (AI for pose estimation)

This example can extend to many other agricultural activities as shown in Fig. 1.8. These three elements are the points that this thesis seeks to address using artificial intelligence.

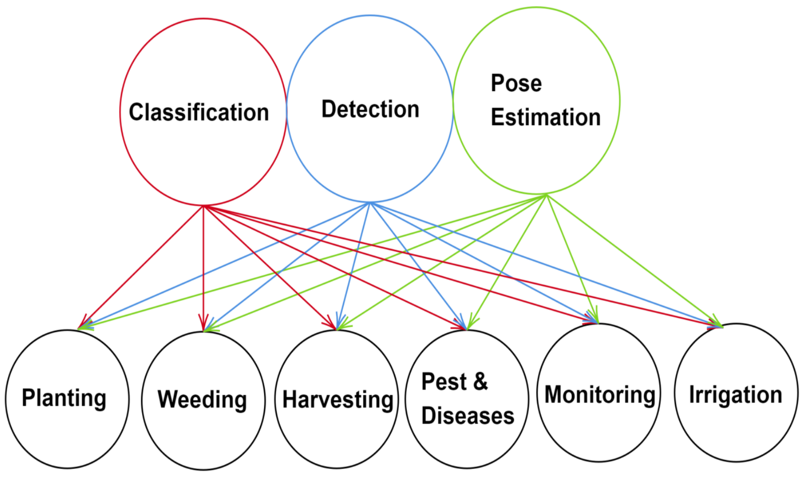


Fig. 1.8 Other Agricultural Activities that Utilises the Three Elements (Classification, Detection and Pose Estimation).

By extension, tackling these elements using AI can provide a pathway to automation in activities such as planting, weeding, harvesting, pest and disease control, crop monitoring, irrigation, etc. Being the core activities in agriculture, this automation, in turn, tackles the problem of sustainability, efficiency, cost, labour scarcity and productivity. In a nutshell, the objectives of this research can be summarised as follows:

- To increase agricultural productivity by the adoption of affordable robotics platforms equipped with AI tools for classification-related agricultural tasks in real-time deployment while possessing better accuracy.

- To increase agricultural productivity by the adoption of affordable robotics platforms equipped with AI tools for detection-related agricultural tasks in real-time deployment while possessing better accuracy.

- To increase agricultural productivity by the adoption of affordable robotics platforms equipped with AI tools for pose estimation-related agricultural tasks in real-time deployment while possessing better accuracy.
- To tackle the problem of robotics teleoperation by utilising a virtual environment thus combining the above-mentioned for other activities like fruit picking in precision agriculture (as a case study).

Remark 1 ***Research Questions***

- *Question 1: Can a robot perform autonomous classification for agricultural applications in real-time? if yes how?*
- *Question 2: Can a robot perform autonomous detection for agricultural applications in real-time? if yes how?*
- *Question 3: Can a robot perform autonomous pose estimation for agricultural applications in real-time? if yes how?*
- *Question 4: Can a robot perform be intelligently teleoperated from a virtual environment for fruit picking? if yes how?*

Chapter 2

Theoretical Background

Chapter abstract

This chapter highlights the background knowledge of the tools and methods used in this research. The fundamental theoretical background that this work builds upon is discussed to familiarise the reader with the concepts behind this work. Camera calibration as a tool for obtaining camera parameters is discussed. Basic principles of artificial intelligence is briefly discussed including regularisation techniques and datasets. The structure of the thesis is also highlighted in this chapter.

2.1 Camera Calibration

This work uses multiple camera sensors. To accurately utilise a camera sensor, the camera needs to be calibrated. Geometric camera calibration, also known as camera resectioning, determines a camera's lens and sensor's specifications. These attributes can be used to correct lens distortion, measure an object's size in world units, or localise a camera. Applications like machine vision use these tasks to facilitate accurate classification, detection and localisation of objects. This technique is also utilised in robots to facilitate navigation and 3-D scene reconstruction. Intrinsic, extrinsic, and distortion coefficients are the camera parameters obtained from calibration. The 3-D world points and the matching 2-D image points are required to estimate the camera settings. These correspondences can be obtained by comparing many pictures of a calibration pattern, such as a checkerboard. Thus, the camera parameters are obtained by using the correspondences. The model proposed by Jean-Yves Bouguet is the foundation of the pinhole calibration algorithm used in this work (Bouguet, 2004). The model contains lens distortion and the pinhole camera model, respectively. Since an ideal pinhole camera does not have a lens, the pinhole camera model does not take lens distortion into consideration. A comprehensive algorithm of the camera model should incorporate radial and tangential lens distortion in order to faithfully simulate a real camera. A pinhole camera has a simple setup with a small aperture and no lens. When light passes through the aperture, it projects an inverted on the other side of the camera. You can imagine the upright image of the scene is on the virtual image plane, which is in front of the camera as seen in Fig. 2.1.

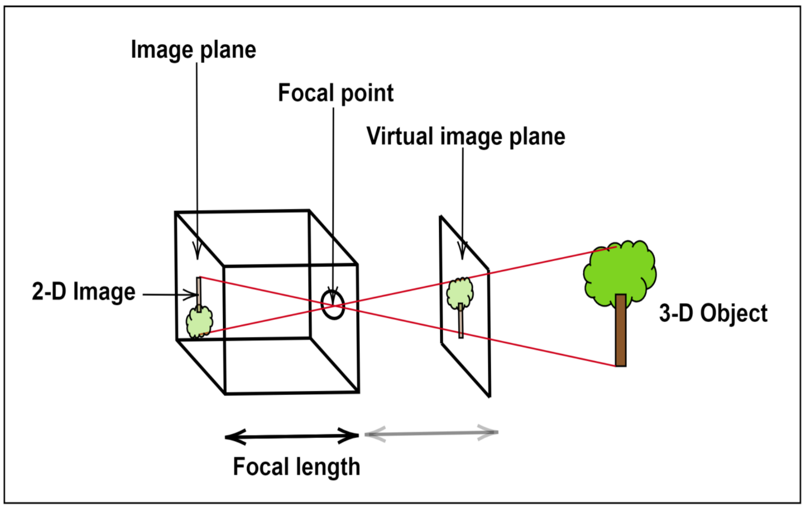


Fig. 2.1 A Simple Pinhole Camera Model

The camera matrix is a 3×4 matrix that represents the pinhole camera specifications. The 3-D world scene is mapped into the image plane via this matrix. The extrinsic and intrinsic parameters are used in the calibration procedure to compute the camera matrix. The extrinsic parameters represent where the camera is located in the three-dimensional scene. The camera's optical centre and focal length are referred to as the intrinsic parameters. Thus, 3-D and 2-D conversions can be performed using equation 2.1

$$W \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = P \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \tag{2.1}$$

and,

$$P = K \begin{bmatrix} R & t \end{bmatrix} \tag{2.2}$$

where, W is the scale factor, $[x, y, 1]^T$ is the image points, $[X, Y, Z, 1]^T$ is the world points. P is the camera matrix. K is the intrinsics matrix. R and t are the rotation and translation which are the extrinsic parameters. The intrinsic parameters can be used to map the camera coordinates into the image plane using the intrinsic parameters. The world points can also be transformed into camera coordinates using extrinsic parameters as shown in Fig. 2.2.

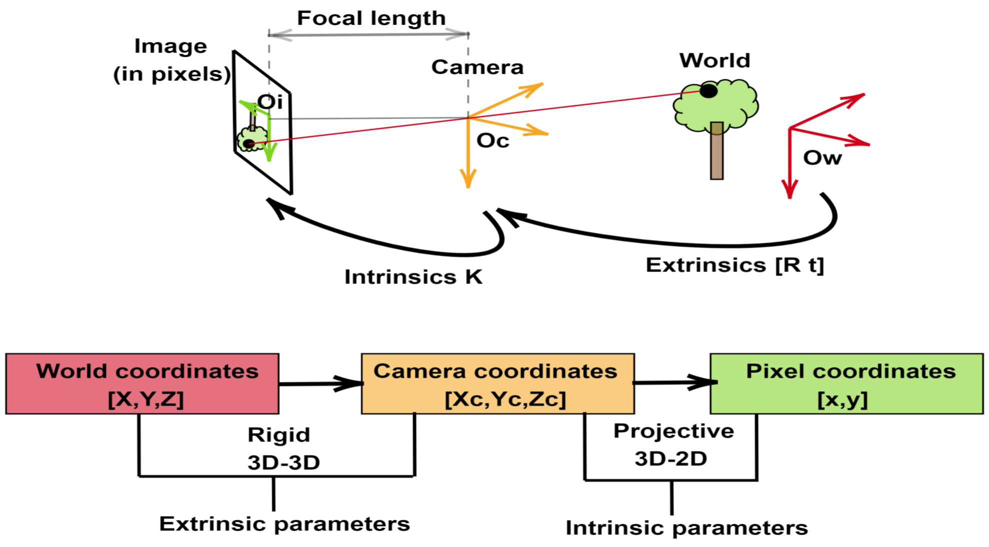


Fig. 2.2 Conversion from world plane to image plane

The calibration algorithm computes the camera matrix by utilising using the extrinsic and intrinsic parameters. The extrinsic parameters represent a rigid transformation from a 3-D world coordinate system to a 3-D camera's coordinate system. The intrinsic parameters represent a projective transformation from the 3-D camera's coordinates into the 2-D image coordinates. The extrinsic parameters encompass the rotation, R, and a translation, t. The origin of the camera's coordinate system is at its optical centre and its x and y-axis define the image plane as seen in Fig. 2.3.

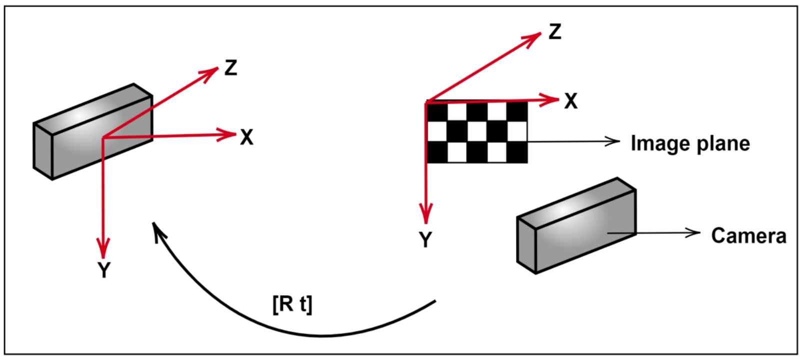


Fig. 2.3 Camera coordinate system

The optical centre, also known as the principal point, focal length, and skew coefficient make up the intrinsic parameters. The camera intrinsic matrix, K, is denoted as:

$$\begin{bmatrix} f_x & s & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \tag{2.3}$$

where (c_x, c_y) is the optical center, (f_x, f_y) is the focal length in pixels and s represent the pixel skew. Pixel skew refers to when the pixels in an image are not aligned properly, resulting in a slanted or skewed appearance. Fig. 2.4 shows an illustration of pixel skew.

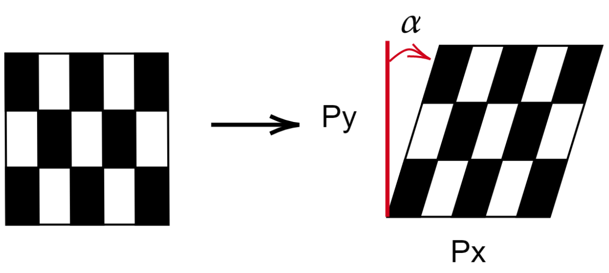


Fig. 2.4 Pixel Skew

Remark 2 ***Basic camera parameters***

$[c_x, c_y]$ - optical center also known as principal points in pixels

(f_x, f_y) - Focal length in pixels

$fx = F/p_x$

$fy = F/p_y$

F - Focal length in world units, expressed in millimetres. (p_x, p_y) - Size of pixels in world units

$s = f_x tan\alpha$

As earlier stated, the pinhole camera model does not account for distortion as represented by the camera matrix. This is because a pinhole camera does not have any lens. However, a real camera model has a lens, and thus distortion has to be accounted for. There are two common types of distortion, the tangential and radial distortion. Tangential distortion occurs when the image plane is not parallel with the lens. The tangential distortion coefficients are utilised for the modelling of this distortion. Consider a distorted point denoted as $(x_{distorted}, y_{distorted})$.
Also,

$$x_{distorted} = x + [2t_1xy + t_2(r^2 + 2x^2)] \tag{2.4}$$

$$y_{distorted} = y + [t_1(r^2 + 2y^2) + 2t_2xy] \tag{2.5}$$

where x and y are the undistorted pixel points which are in normalised image coordinates. Normalised image coordinates are computed from the pixel coordinates by translating the optical centre and dividing by the focal length which is both in pixels and resulting in a dimensionless unit. Also, $r^2 = x^2 + y^2$, t_1 and t_2 are the tangential distortion coefficients of the camera lens.

Radial distortion is said to happen when light rays bend near the edges of the lens more than they do at the optical centre. Smaller lens often results in greater distortion and vice-versa. This type of distortion is modelled with the radial distortion coefficients. Consider a

distorted point denoted as $(x_{distorted}, y_{distorted})$.
Also,

$$x_{distorted} = x(1 + m_1 r^2 + m_2 r^4 + m_3 r^6) \tag{2.6}$$

$$y_{distorted} = y(1 + m_1 r^2 + m_2 r^4 + m_3 r^6) \tag{2.7}$$

where x and y are the undistorted points and are dimensionless as in the case of tangential. $r^2 = x^2 + y^2$, m_1, m_2 and m_3 are the radial distortion coefficients of the camera lens.

2.2 Artificial Intelligence

Artificial intelligence is becoming an established subject in science, technology, mathematics and engineering disciplines. Although there are many definitions of what this subject might be, the commonest and most relatable is the human-centric one, which is the ability to perform tasks as humans and in fact better than humans do. However for systems to be referred to as intelligent (Stuart and Peter, 2013) propose that they should pose some characteristics which are Natural language processing for communication, representation of knowledge to store what the system knows or learns, reasoning to use the stored information to answer questions and provide new information, and the ability to recognize and extrapolate patterns to learn novel information.

2.2.1 Machine Learning

Machine learning (ML) is a branch of artificial intelligence that automatically processes data. The availability of voluminous datasets in the modern days makes it pertinent for machine learning techniques to adapt and maximise the potential of such datasets.

Generally, ML can be categorised into two central types: supervised and unsupervised learning (Kevin, 2012). The objective of a supervised type of learning is to learn the relationship or map from the inputs to the outputs given labelled pairs. Unsupervised learning, on the other hand, has the objective of recognising patterns in the input data in the absence of any labels or outputs for comparison. This type of learning is naturally more applicable, just as in the case of humans that can learn by simple visual observation. Nonetheless, this type of learning is not properly defined and thus encounters more frequent challenges than the former. Moreover, computer vision is the closes machine analogue for object perception and the robotics domain is also one discipline that utilises AI properly. The flexibility of robotics platforms coupled together with the availability of small compact and affordable cameras makes them the favourite candidate for computer vision applications. Raw, rich information about the environment can easily be captured for input data in robotics applications.

Normally, machine learning encompasses two stages. The first stage is the feature extraction stage. This stage involves extracting discriminative and informative subsets from supplied raw data. In the context of this work, relative pose estimation utilises the data from a camera to obtain the relationship between the image points and the scene and thus requires the extraction of meaningful features or points from the image. Also, weed classification leverages the extraction of features from weed images captured with the camera. The second stage rallies around selecting a befitting model to process the extracted features or points to give the desired results. These models are often generated by the ability to minimise the prediction error.

2.2.2 Deep learning

Artificial neural networks (ANNs) as part of deep learning is a branch of ML in which models try to learn directly from the supplied raw input data. Inspired from biology of the brain's function (Goodfellow et al., 2016), the basic model is the Multilayer Perceptron (MLP) (Christopher, 2006) to generate a vector of outputs $y = [y_1 \dots y_K]^T$, M linear combinations of the elements of the input $x = [x_1 \dots x_D]$ are established and the result is passed through a nonlinear activation function h:

$$a_j = \sum_{i=1}^{D} W_{j,i}^{(1)} x_i + b_j^{(1)}, j = 1, \dots, M \tag{2.8}$$

$$z_j = h(a_j), \tag{2.9}$$

Where $W_{j,i}, b_j$ are the network's learnable parameters (weight and biases), and the superscript (1) represents the first layer of the MLP. In the case of only one hidden layer, through another linear combination, the output activations can be directly obtained from z_j

$$a_k = \sum_{j=1}^{D} W_{k,j}^{(2)} z_j + b_k^{(2)}, k = 1, \ldots, K \tag{2.10}$$

The output activations can be forwarded through another activation unit or taken as the identity $y_k = a_k$ depending on the nature of the responses as shown in Fig. 2.5.

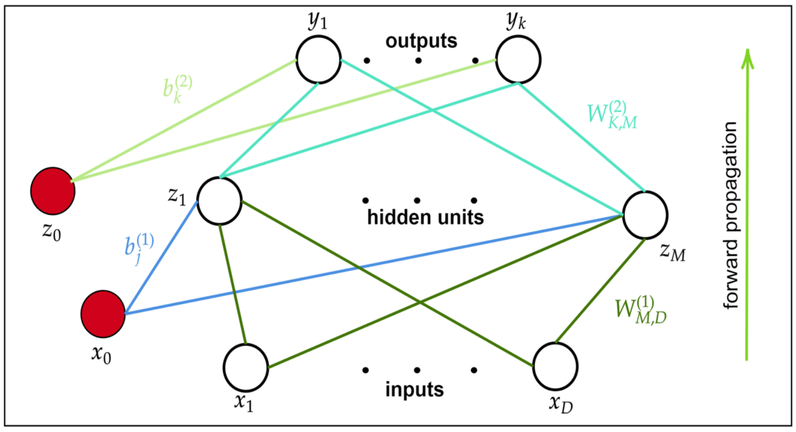


Fig. 2.5 Diagram for a two-layer artificial neural network (ANN). The open nodes represent input (x), hidden (z), and output (y) variables.

As more development continues in the optimisation of such models coupled together with the invention of high computing capabilities mainly the graphics processing units (GPU), models having larger depth i.e having a large number of hidden layers can be trained in a shorter time frame. This gave rise to enormous research in the deep neural network (DNN) field.

The training of the DNN is achieved as a minimisation of a scaler loss function Like in classical ML, the training of a DNN is formulated in terms of minimising a scalar loss function $f(x, y, \lambda)$, where λ is the set of network's learnable parameters and the dependency on the inputs and outputs has been made explicitly. Since the model is nonlinear by design (Eq. 2.9), the optimal λ can be achieved using iterative methods to search for critical points in f with the gradient information. The simplest method is through gradient descent (Christopher, 2006), by taking a small step in λ-space in the negative gradient direction.

$$\lambda^{(k+1)} = \lambda^{(k)} - \alpha_k \Delta_\lambda f, \tag{2.11}$$

where α_k is the learning rate at time-step $\tau = \tau_k$. The gradient f with respect to the weights for each layer can be obtained from local gradients of the respective layer by using backpropagation (Rumelhart et al., 1985). Consider a unit j in one layer that sends connections to k units in the next layer, the local gradient at unit j is obtained as:

$$\frac{\delta f}{\delta a_j} = \sum_k \frac{\delta f}{\delta a_k} \frac{\delta a_k}{\delta a_j} \tag{2.12}$$

Due to the memory constraint of gradient-based parameter optimisation for voluminous datasets, a minibatch $\mathbb{B} = \{x^{(1)}, \ldots, x^{(m)}\}$ of m inputs at each step can be sampled from the training data and utilised instead. Thus, the process results in a stochastic gradient descent (SGD) and can be estimated as:

$$\Delta_\lambda f \approx \frac{1}{m} \sum_{i=1}^{m} \Delta_\lambda f(x^{(i)}, y^{(i)}, \lambda) \tag{2.13}$$

Typically a model will require many epochs to train. One epoch will mean that the SGD has processed all the mini-batches consisting of the entire datasets of inputs. A linear output as portrayed in Eq. 2.10 allows the network to learn regression problems. More traditionally, deep learning is often associated with classification problems. i.e. discrete and mutually exclusive output. The sigmoid function can be utilized to model an output that obeys the Bernoulli distribution. The softmax function is utilised to model categorical or generalised Bernoulli distributions and is given by:

$$softmax(a)_k = \frac{exp(a_k)}{\sum_l exp(a_l)} \tag{2.14}$$

2.2.3 Activation Functions

The hyperbolic tangents and the sigmoid were the commonly used activation functions in the early days of ANNs (Goodfellow et al., 2016) (see Fig. 2.6, left and middle respectively). However, the sigmoid function is sensitive to inputs close to zero and thus is susceptible to vanishing gradient problems in backpropagation and renders training difficult. The hyperbolic tangents also depend on small activation inputs but are generally easier to train. Thus, modern-day research works often choose the rectified linear unit (ReLU) as the default activation function (see Fig. 2.6, right). It is defined as follows:

$$relu(x) = max(0,x) \tag{2.15}$$

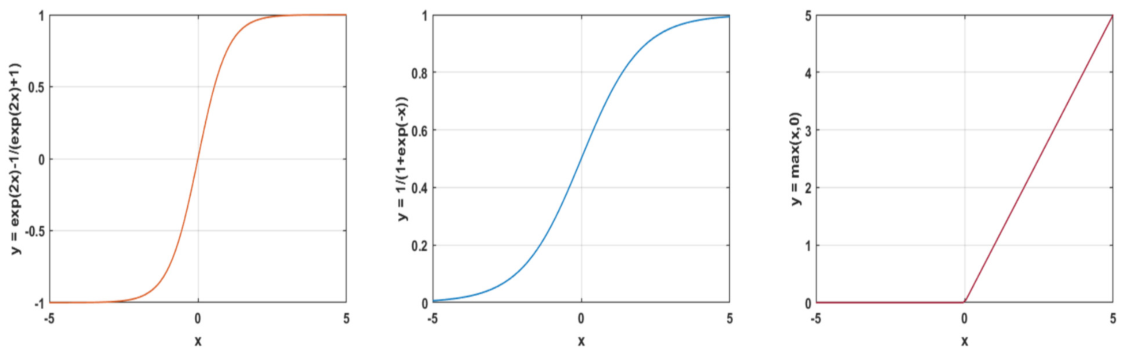


Fig. 2.6 Activation functions for ANN. left: Hyperbolic tangent function. middle: Sigmoid function and right: Rectified linear unit.

The ReLU function gives an output of zero when the input is zero and has a linear response input if the input is positive. Another advantage of the ReLU is sparse activation which prevents overfitting and thus performs well with new examples. The hyperbolic tangent functions and sigmoid saturates in both directions thus, the ReLU has better gradient propagation when compared to them.

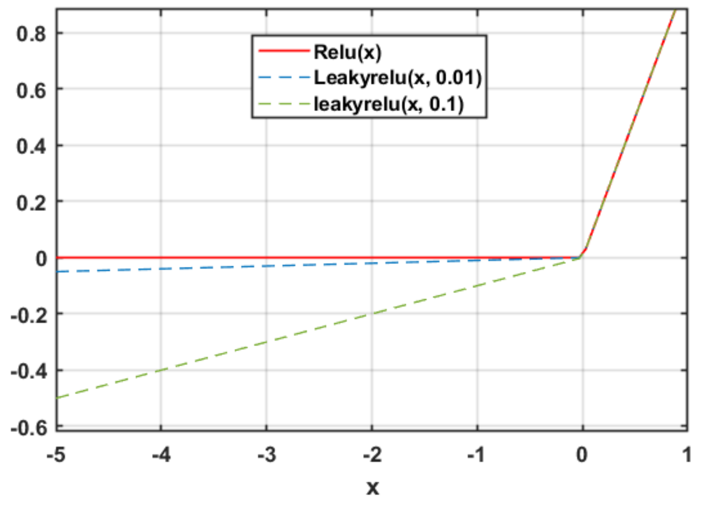


Fig. 2.7 Comparison between the rectified linear unit (ReLU) activation function and the leaky ReLU activation function. Leaky ReLU is plotted with different values of ϕ

Remark 3 ***Dying ReLU***

As mentioned earlier, the ReLU has a superior advantage over the rest activation functions. However, it faces a problem during backpropagation. This problem is referred to as dying ReLU and regardless of the input, it has a nature of converging to some states of inactivity. It happens when the network activations give an output of zero value due to learning a large negative bias. Over the years, some solutions have been proposed to mitigate this problem such as the leakyReLU (Maas et al., 2013) that gives a small gradient when the unit is not active as seen in Fig. 2.7. Defined as follows:

$$leakyrelu(x,\phi) = \begin{cases} x & \text{if } x > 0 \\ \phi x & \text{otherwise} \end{cases} \tag{2.16}$$

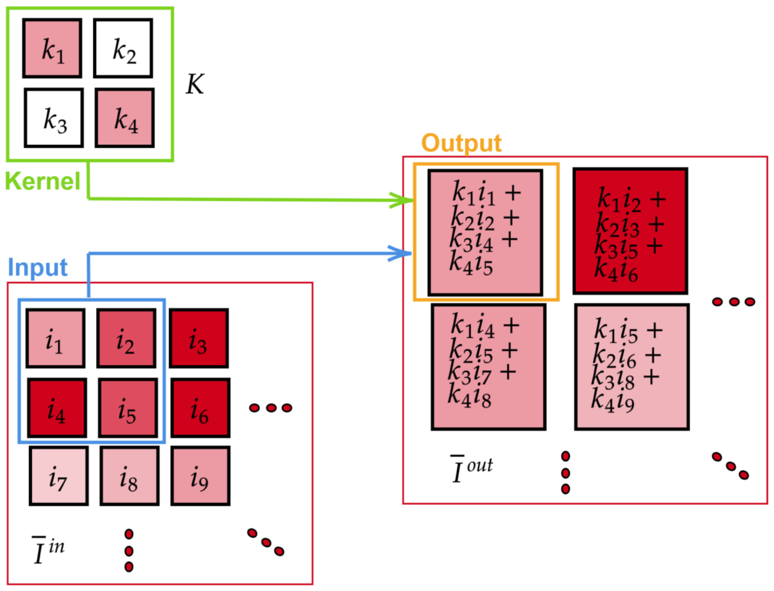
2.2.4 Convolutional Neural Networks

Image-based applications benefit from the end-to-end nature of DNNs due to the ability to learn the optimal feature representation in an unsupervised way. An example of this application is in DNN-based pose estimation method. The model is capable of understanding the nonlinearities between the input images and the 6-DOF pose estimations. The hidden layers facilitate the linear combination between all the possible inputs and outputs. Thus, each layer is referred to as a fully connected (FC) layer. Even for an average-resolution image with fewer pixels, training an FC layer would require a very high computing power as each pixel will be an input to the DNN, alternatively, a convolutional neural network (CNN) can be used to process such types of image inputs (Le Cun et al., 1989).

A convolution operation is performed by sliding a 2D kernel K which is a relatively small square matrix having an odd dimension across an image $\bar{I}^{in}$ that is 2D to produce an output $\bar{I}^{out}$ that is also 2D as follows:

$$\bar{I}^{out}_{i,j} = (\bar{I}^{in} \times K)_{i,j} = \sum_u \sum_v \bar{I}^{in}_{i+u,j+v} K_{u,v} \tag{2.17}$$

Fig. 2.8 shows a convolution operation of a 2D input image with a 2×2 kernel to yield the desired output.



k_1 k_2 k_3 k_4
K
Kernel
Output
Input
i_1 i_2 i_3 i_4 i_5 i_6 i_7 i_8 i_9
$\bar{I}^{in}$
$k_1 i_1 + k_2 i_2 + k_3 i_4 + k_4 i_5$
$k_1 i_2 + k_2 i_3 + k_3 i_5 + k_4 i_6$
$k_1 i_4 + k_2 i_5 + k_3 i_7 + k_4 i_8$
$k_1 i_5 + k_2 i_6 + k_3 i_8 + k_4 i_9$
$\bar{I}^{out}$

Fig. 2.8 A two-dimensional convolution process. An input image $\bar{I}^{in}$ is convolved with a 2×2 kernel K to produce an output image $\bar{I}^{out}$. This illustration is only shown for the top left 3×3 sub-matrix of $\bar{I}^{in}$.

The convolutional layers have lesser memory requirements as compared with the FC layer. This is due to a decrease in parameters and the sparse nature of the learnable parameters.

Regardless of the input location of the parameters, the parameters can be learned considering that the kernel is slid over the entire image. The resultant output plays the function of a feature map, representing the location and intensity of localised features, which has the same impact as looking for localised features in an image. In fact, CNNs are widely employed as feature extraction front-ends, where more feature maps are generated while spatial information is sequentially reduced. In this instance, the feature maps appear as images with multiple channels obtained from the convolution and extending the 2D kernel to a 4D tensor κ having a dimension of:

$$dim\kappa = C_{in} \times (F \times F) \times C_{out} \tag{2.18}$$

where C_{in} is the number of input channels, C_{out} is the number of output channels (desired) while assuming that the kernel is spatially square with a dimension of $F \times F$. Pooling operations can be performed after the convolution for spatial reduction where bins are generated by sub-dividing the feature map of spatial dimension W_{in} to obtain an output of the following dimension:

$$W_{out} = \left[\frac{W_{in} - Q}{Q} \right] + 1 \tag{2.19}$$

Q is the pooling window size. The pooling operations usually take either the average or maximum value of each bin. In most of the recent DNN models, increasing the stride S of the convolution is preferred over pooling layers. The stride refers to the number of rows and columns that are skipped while sliding the kernel. Thus, the spatial output size, in this case, can be formulated as follows:

$$W_{out} = \left[\frac{W_{in} - F + 2P}{S} \right] + 1 \tag{2.20}$$

where P is the spatial padding that was applied to the input.

2.2.5 Transfer Learning

DNNs that result in sophisticated models may necessitate not only lengthy training time but also larger training datasets. To overcome these two challenges, Transfer learning, which makes the assumption that some factors impacting the outcome of one task are relevant to the outcome of another task is an approach that can be utilised. For CNN for instance, it is expected that the multiple learned kernels will always converge to detect general visual features. Even experimentally, it has been verified that tasks such as image classification optimise the kernels of the beginning CNN layers towards a wider domain and broad features such as edges and corners, while the kernels of the last or ending layers are more critical in problem-specific shapes (Zeiler and Fergus, 2014). As a result, it is typical to use a pre-existing CNN architecture as a model backbone, with the early layers trained on a broad and general dataset up to a point. The final few layers or newly added layers can then be trained from the ground up using smaller domain-specific data. The ImageNet dataset consists of more than 14 million images that are manually classified into bins of more than 20,000 categories via crowd-sourcing. It is a dataset that is widely used in transfer learning for visual tasks. The ImageNet Large Scale Visual Recognition Challenge (ILSVRC), whose goal is to construct a ML pipeline to accurately classify images into 1000 different classes, served as an inspiration for a lot of advancement in CNN architectural design. Since then, new CNN architectures have participated in the challenge every year, leading to significant advancements while showcasing new innovations, a notable example is the ResNet (He et al., 2016a), which proposed residual connections allowing innovations that led to even deeper architectures. Another example is GoogLeNet (Szegedy et al., 2015) which is associated with a very deep architecture as well as parallel layers to detect features at different scales. The majority of these cutting-edge DNN architectures are open-sourced, with pre-trained ImageNet weights made available; this has aided the quick development in this field and adoption of models like CNN front-ends.

2.3 Regularisation

The capacity of the model, also known as the number of learnable parameters in DNNs, is an implicit hyperparameter. The potential for the DNN to make predictions accurately increases with capacity. Overfitting, on the other hand, is when the network really stops learning and rather memorises the training data, making it impossible for it to generalise to new, untested test data, and is caused by having too many parameters. Due to this factor, it is common practice to periodically evaluate the network's performance on a validation set while it is being trained. A validation set is not utilised in training the model but rather is used to simply shed light on the pipeline's generalisation capabilities. Overfitting occurs especially when the training error is substantially smaller than the validation error. On the other hand, fewer parameters can lead to underfitting which results in both high validation and training errors. Overfitting can be tackled by using fewer parameters, and early stopping, which involves stopping the training process when the validation error begins to diverge from the training error. Furthermore, regularisation techniques can be introduced, which essentially adds noise to the learning process. Unless the training dataset comprises a very large number of examples in millions, regularisation should generally always be incorporated (Goodfellow et al., 2016). A few common regularisation techniques for DNNs are outlined in this subsection.

2.3.1 Dropout

Dropout is the process of randomly removing units from a network by multiplying their output value by zero (Hinton et al., 2012). For a very deep network, Dropout makes an attempt to mimic the ML notion of bagging which means training K separate models for K separate training data subsets. A binary mask is randomly sampled for each minibatch step and applied to the hidden units of each layer with probability p. This probability is fixed and used as a layer's hyperparameter. Before the activation function, typical values for FC layers are $p = 0.5$ while for convolutional layers are $p \in [0.1, 0.2]$.

2.3.2 Weight Decay

One of the pioneering regularisation techniques used in machine learning, even before ANNs, is weight decay. It entails introducing a term into the loss function that is proportional to the weights of each layer. L^2 regularisation is a standard sort of weight decay that adds the squared sum of the weights only without biases as they are normally disregarded in weight decay. Despite the lengthy existence of weight decay in the field of ML, different techniques are often used in addition to or as a whole replacement for modern CNNs.

2.3.3 Batch Normalisation

In order to normalise the inputs to a layer, (Ioffe and Szegedy, 2015) computes the mean $\mu_{\mathbb{B}}$ and variance $\sigma^2_{\mathbb{B}}$ for each mini-batch m. The mean and variance of the complete dataset are approximated throughout the training phase by calculating the moving average of each per-batch $\{\mu_{\mathbb{B}}, \sigma^2_{\mathbb{B}}\}$, which is then utilised to normalise inputs. Overall, the batch norm was designed to enhance DNN optimisation, however, it introduces noise into the system, which may result in a regularising effect.

2.3.4 Image Augmentation

Image augmentation is a powerful regulariser that generates extra (augmented) data while indirectly training a DNN on which features are crucial to learning. For example, in the case of CNNs, the performance of a classifier could be increased simply by randomly rotating and flipping the input image in-plane, boosting the robustness of classification and detection.

2.4 Datasets

A dataset is a collection of coherent ingredients that accurately depict the reality in which the method wants to be verified and evaluated, rather than a collection of isolated, random information elements. Datasets play an important role in the evaluation of an algorithm. An ideal dataset samples information straight from that same reality. A good dataset also allows for a fair evaluation of an algorithm's performance as compared to other algorithms under the same conditions (benchmarking). As interest in machine learning is increasing, so does interest in raw data as these datasets are required for the training procedure of the models. Thus, there is a need for more open-source datasets which should ideally be well structured, labelled and depict the clear ground truth of the scenario so that algorithms can be effectively trained to detect, classify, estimate and evaluate.

In the context of this thesis, datasets are very crucial for the intelligence of robotics platforms for agriculture. This work utilised both open-sourced and custom-made dataset to teach the models and evaluate the outcome. For example, weed classification and detection is a common problem in the precision agriculture domain and thus the availability of labelled datasets for this purpose is heterogenous. Although weeds vary from one geographical region to another, the aim is to prove the efficiency and performance of the models in general. Shortage of dataset for other crucial activities such as fruit 6D pose estimation led to the generation of a custom dataset that tallies with the purpose of this work. Along the thesis, the dataset utilised for the experiments are elaborated upon.

2.5 Outline and Contributions

The structure of the thesis is presented as follows:

- **Chapter 1** introduces this work while highlighting the research context and motivation of this thesis.
- **Chapter 2** provides a brief introduction on the theoretical background, and structure of the thesis.
- **Chapter 3** proposes the use of a novel artificial intelligence-based solution for weed detection and classification on robotic platforms. This chapter aims to achieve real-time detection and classification of weeds so that they can be precisely sprayed and removed thereby avoiding the usage of large quantities of harmful chemicals in the environment.
- **Chapter 4** presents a novel monocular vision-based approach for drones to detect multiple types of weeds and estimate their positions autonomously for precision agriculture applications. This chapter builds on the pipeline presented in Chapter 1 and extends it to not only detection and classification but also relative position estimation. This can reduce the labour deficit in the sense that a UAV can patrol over a large area

and simultaneously detect and estimate the position of weeds. An extension of this is to communicate the estimated positions to a ground robot which will then utilise such positions to navigate and perform the required action on the weeds.

- **Chapter 5** proposed TransPose, a novel improved Transformer-based 6D pose estimation with a depth refinement module in the quest for better performance on 6D pose estimation. This is a persistent problem, especially in precision agriculture. The architecture incorporates an innovative lighter depth estimation network architecture that estimates depth from an RGB image using the feature pyramid method to refine the pose regressed from a transformer model. Another major contribution is tailoring the pipeline for fruit picking applications by utilising a first-ever multi-modal custom-made fruit dataset tailored specifically for fruit 6D pose estimation which is called *"Fruity"*. The result obtained competes with the state-of-the-art methods.

- **Chapter 6** presents a method that utilises virtual reality as a tool for an immersive experience in a 3D environment which allows users to visualise and interact with robots in such environments. The pipeline proposed is a seamless teleoperation pipeline that interfaces virtual reality with robots for precision agriculture tasks aiming at achieving remote operations. This chapter builds on the work in Chapter 4 and incorporates it into the framework to estimate the 6D pose of the target fruit in real-time and utilises this information to render the target fruit in the virtual environment which is then grasped or harvested. Another contribution of the pipeline is the flexibility it offers as it has also been successfully deployed on other robotics platforms.

- **Chapter 7** concludes this thesis while discussing the results and proposing future works that can build on this thesis.

2.6 Published and Submitted Manuscripts

Conferences

- Abdulsalam, M., & Aouf, N. (2020, September). Deep weed detector/classifier network for precision agriculture. In 2020 28th Mediterranean Conference on Control and Automation (MED) (pp. 1087-1092). IEEE.
- Abdulsalam, M., & Chekakta, Z., Hogan, M., & Aouf, N. (2023, June). Fruity: A Multi-modal Dataset for Fruit Recognition and 6D-Pose Estimation in Precision Agriculture. In 2023 31st Mediterranean Conference on Control and Automation (MED) (pp. 144-149). IEEE.
- Abdulsalam, M., & Aouf, N., (under review) VitRob Pipeline: A Seamless Teleoperation Pipeline for Advanced Virtual Reality - Robot Interface Applied for Precision Agriculture. In the 2023 IEEE International Conference on Robotics and Biomimetics (ROBIO).

Journals

- Abdulsalam, M., Ahiska, K., & Aouf, N. (2023). A novel UAV-integrated deep network detection and relative position estimation approach for weeds. Proceedings of the Institution of Mechanical Engineers, Part G: Journal of Aerospace Engineering. Sage Journal.
- Abdulsalam, M., & Aouf, N. (under review). TransPose: A Transformer-based 6D Object Pose Estimation Network with Depth Refinement. Engineering Applications of Artificial Intelligence. A journal of IFAC, the International Federation of Automatic Control. Elsevier.

Chapter 3

Deep Weed Detector/Classifier Network for Precision Agriculture

Chapter abstract

This chapter discusses the development of an artificial intelligence-based solution for weed detection and classification on robotic platforms. The productivity of crop farming keeps diminishing at an alarming rate due to the infestation of weeds and pests. Deep learning is becoming the approach for identifying weeds on farmlands. However, training weed data sets with deep learning for classification alone trains the whole image consisting of the weed and its background (soil) without categorically telling which particular item in the image is a weed. This makes the adoption of this classification approach for precision agriculture difficult. Thus, the chapter presents an alternative approach, which involves incorporating a pre-trained network by adopting the ResNet-50 framework and YOLO v2 object detector for weed detection/classification on farmlands. Thus, weeds can precisely be located, identified (type) and eventually sprayed with the appropriate herbicide or removed with the appropriate mechanism. This sums up the weeding process in precision agriculture.

3.1 Motivation

Weed control is an important issue in agriculture (Pusphavalli and Chandraleka, 2016a). The conventional method used for weed control involves treating the entire field uniformly with a single herbicide dose. Thereby, spraying the soil, crops, and weed in the same manner (Lottes et al., 2017a). This approach is easier and faster. However, spraying the herbicide uniformly might not necessarily work for every weed as there are different species. Also, a large amount of chemicals is wasted on the soil as this spraying process is not targeted at a particular weed. An important objective in weed management is discrimination between weed species to control each specie using the appropriate herbicides. Efficiency is higher if selective treatment is performed for each type of infestation instead of using a broadcast herbicide on the whole surface.

Precision farming provides a new solution using a systematic approach for today's agricultural issues such as the need to balance productivity with environmental concerns (Hakkim et al., 2016). Precision agriculture involves the adequate and optimum usage of resources based on various parameters governing crop yield (Satish Kumar and Sudeep, 2007). The Handbook of Precision Agriculture (Srinivasan, 2006) describes precision agriculture as a holistic and environmentally friendly strategy in which farmers can vary input use and cultivation methods – including the application of seeds, fertilizers, pesticides and water, variety selection, planting, tillage, harvesting – to match varying soil and crop conditions across a field.

3.2 Related Work

For more precise farming, robots can perform targeted weed treatment if they can specifically locate the position of the unwanted plant and detect the type of weed. This will reduce by a large margin the use of agrochemicals on farms and favour sustainable agriculture. Our approach in this work is useful in the use of chemicals in the right quantity, in the right place and at the right time. Several works have been done in this direction. The work of (Lottes et al., 2018a) focused on weed detection using semantic segmentation using a sequenced approach. (Zheng et al., 2019a) suggested the merging of networks for Crop detection. Weed detection performed by (Yu et al., 2019a) concluded on several networks for detection of individual weeds peculiar to perennial ryegrass. (Pusphavalli and Chandraleka, 2016a) worked on an automatic weed removal system using images. (Lottes et al., 2017a) did an impressive work on UAV−Based crop and weed classification for smart farming. (Lottes et al., 2018a, 2017a) generalized all weeds as *’weeds’* without categorically telling what type. While (Zheng et al., 2019a) worked on crops, (Yu et al., 2019a) VGGNet have been proven to have lower accuracy in the work of (Zheng et al., 2019a). Weed detection in precision agriculture is beyond identifying weeds only, it should also extend to telling the type of weed identified so that accurate and precise measures can be taken to remove the weeds. Weeds have variable characteristics depending on the type of weed. Consequently, not all herbicides can work on all weeds and also not all removal mechanisms can work as some weeds have broad leaves and others have embryonic leaves. To be able to tell specifically the type of weed detected will enhance precision and accuracy in removal leading to appropriate measures taken. This chapter presents a novel approach that precisely detects weed and extends to tell precisely what type of weed is detected. The aim is to develop deep learning techniques to detect and classify four classes of weeds which are peculiar but not limited to the corn plant. These weeds are *bluegrass*, *chenopodium*, *circium* and *sedge*. Deep learning allows computational models that are composed of multiple processing layers to learn representations of data with multiple levels of abstraction (LeCun et al., 2015) which can be utilised to detect the intricate structures of weeds such as the patterns on the leaf, the shape of the leaf, the surface areas and edges.

3.3 Deep neural network for weed detection

For more precise farming, robots can perform targeted weed treatment if they can specifically locate the position of the unwanted plant and detect the type of weed. This will reduce by a large margin the use of agrochemicals on farms and favour sustainable agriculture as well as making the task autonomous. Our approach in this work is useful in weed treatment with the right quantity, in the right place, and on the right target weed. we present this novel approach that precisely detects weeds and by extension predicts what type of weed is detected thus assigning a befitting bounding box to the detected weed. A dataset of various weeds is acquired from open source. Weed species peculiar to corn are selected among all the open-source data (Jiang et al., 2020). The dataset consists of four different classes namely Cirsium setosum, Chenopodium album, bluegrass and sedge. The dataset was acquired using a Canon PowerShot SX600 HS camera vertically above the weed to account for reflection. Factors such as illumination and soil background situations are varied to provide a rich and complete dataset. The dataset contains a total of 6000 weed images of size 800 $\times$ 600.

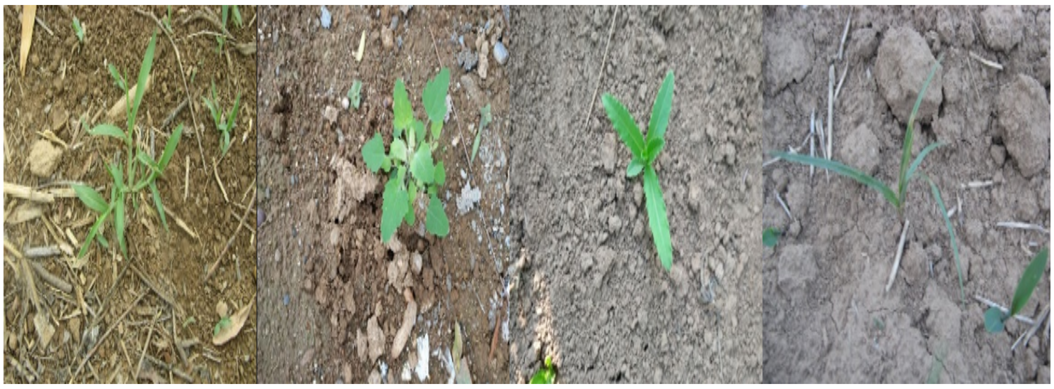


Fig. 3.1 Weed dataset samples. From left-right, Bluegrass, Chenopodium, Circium and Sedge weed

The Bluegrass weed class is a type of weed that affects corn plants. It is one of the most problematic weeds. Leaf blades are smooth with boat-shaped tips and are light green in colour. The Chenopodium class of weed is a fast-growing weed with a width of about 1–3 inches and a length of 3–6 inches. Even though it is edible, it can also be weedy and act as a parasite to the main crop. The circium setosum class is a perennial weed that grows up to $0.5m$ tall. The sedge class of weed is invasive in other types of turf grass and can be very challenging in controlling.

To effectively train a deep network, images are resized to acceptable sizes. The sizes depend on the dimension of the neural network's input layer. Therefore, the images from the dataset are directly resized to 224 $\times$ 224 to fit the input layer while avoiding edge padding.

As with every detection model, ROIs are demarcated on the image dataset. This is often done using an image labelling tool. MATLAB provides a friendly Graphical User Interface (GUI) that allows users to select the ROI on an image in the form of a bounding box as shown in Fig 3.2. The image dataset can simply be imported as a folder containing all the images arranged serially. The classes are then defined and named. The bounding box operation involves drawing a box around the Region of Interest (ROI) such that the ROI is contained within. The bounding box information is exported in the form of a 1×4 vector containing the x and y pixel coordinates of one corner, length and width of the bounding box. This Matrix serves as the ground truth of the labelled object and can easily be exported to the MATLAB workspace for utilisation by the network.

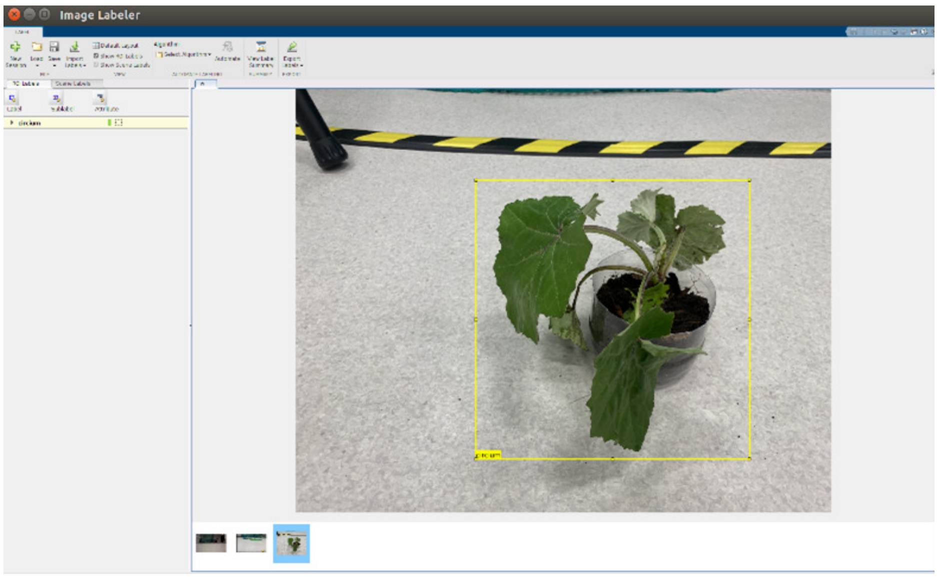


Fig. 3.2 MATLAB Image Labelling GUI demonstrating a labelling process of a weed. The bounding box in yellow colour is drawn manually to enclose the ROI

3.4 Weed Detection Stages

The deep learning technique developed for weed detection/classification is composed of three stages as shown in Fig. 3.3. These stages summarise the procedure of the deep detector/classifier network. The stages can be itemised as follows:

- Input stage
- Processing stage
- Output stage

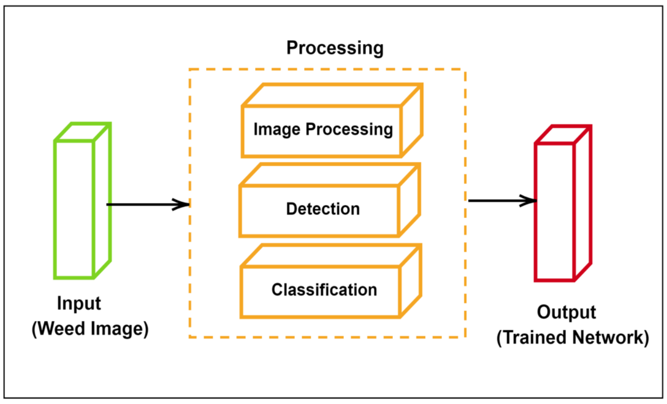


Fig. 3.3 Block diagram of weed detection stages

The input stage consists of the processes associated with the image dataset of the four classes of weeds. A total of 1000 images are sampled from each class of weed for training and 200 images are allocated from each class for testing. The used images are of several sizes, different visual appearances, different soil environments and different growth stages to provide effective training.

The processing stage is subdivided into three sub-stages, namely: image processing, deep learning image classification network and deep learning object detector network as seen in Fig. 3.3. The image processing stage is important because we resize the image size to an acceptable size that is compatible with our network. The acceptable size is 224×224. Furthermore, the objects to be detected needs to be labelled on the image. The ROI pixel and the bounding box information are stored. The images with their corresponding label now serve as the input to the deep-learning image classification network.

The classification network adopts a deep learning technique to train images to classify outputs into several classes. ResNet-50 is used as the classification network. A second network is incorporated into the overall model pipeline. This network develops a deep object detection algorithm. Object detection is performed by isolating a certain object of interest within the image and determining where they are located on the image. You Only Look Once (YOLO) framework is imbibed for this purpose. Finally, the output stage consists of our trained hybrid network. The deep weed classifier/detector network can classify and detect the four classes of weeds.

3.5 ResNet-50 as a Weed Classifier Network

As reported in the work of (Zheng et al., 2019a), deep convolutional neural networks have become the dominating approach for image classification. Many architectures such as VGG (Simonyan and Zisserman, 2014a), Densenet (Huang et al., 2017) and SqueezeNet (Iandola et al., 2016) have given very impressive performances in image classification. However, adopting or developing a deep architecture to solve realistic problems requires that some aspects should be taken into consideration such as the type or number of layers. A higher number of parameters of these layers increases the complexity of the system and directly influences the memory, the computation, the speed and the results of the system (Zheng et al., 2019a).

SqueezeNet is a network developed by DeepScale and has an accuracy similar to that of AlexNet. However, it is a CNN architecture that has 50 times fewer parameters and is compressed to less than 0.5MB, or 510 times smaller than AlexNet without compression. SqueezeNet was developed with ImageNet as a target dataset. DenseNet introduces direct connections between any two layers with the same feature map size. It naturally scales to hundreds of layers, while exhibiting no optimization difficulties (Huang et al., 2017).

The Visual Geometry Group (VGG) of Oxford presented the VGG-16 and VGG-19 (Simonyan and Zisserman, 2014a). VGG network requires an input of $224 \times 224 \times 3$. VGG-16 increases network depth by small convolution kernels for better capacity and makes a good general performance. The VGG-19 is an extension of the VGG-16 where fully-connected layers and pooling layers are added.

The pre-trained deep learning network adopted in this work for the detection/classification of weeds is the ResNet-50 network, which has a total of 177 layers. Deeper neural networks are generally difficult to train. However, for ResNet-50, layers are explicitly reformulated as learning residual functions connecting the layer's input (He et al., 2016b). This means that partial data of the input goes directly to the output without passing through the neural network (Zheng et al., 2019a). This eases the training of deeper networks as some of the original information is preserved. The network learns the differences between the input and the output, or residual instead of learning the desired mapping directly. This residual connection skips over some layers thereby letting the gradient flow through the network easily. This arrangement allows the network to train deeper and the problem of gradient dispersion in backpropagation is perhaps solved.

The residual block of ResNet-50 is defined as follows:

$$\mathbf{y} = \mathscr{F}(\mathbf{x}, \mathbf{W}_i) + \mathbf{x} \tag{3.1}$$

where $\mathbf{x}$ and $\mathbf{y}$ are the input and output feature maps of the residual block, $\mathbf{W}_i$ are the weights of the block, and $\mathscr{F}$ is a sequence of convolutional layers followed by non-linear activations. The addition of the input $\mathbf{x}$ and the output of $\mathscr{F}$ allows the network to bypass some of the convolutional layers, making it easier for the network to learn the residual mapping.

ResNet-18 and DenseNet-121 presented a very high accuracy of about 99.62% and 99.56% respectively in crop classification in the work of (Zheng et al., 2019a). Both networks with accuracy above 99% could not outperform ResNet50 in crop classification. Indeed, ResNet50 is the best-performing model achieving an average accuracy of 99.81% on the CropDeep datasets (Zheng et al., 2019a). In our case, crops, plants and weeds have some major similarities. Thus, this work is inspired to adopt this as a classification backbone.

3.6 YOLO Framework for Weed Detection

YOLO is a single-stage object detection network. It has a fast detection speed and it is trained by dense and regular sampling over locations, scales, and an end-to-end flow. YOLO simplifies weed detection as a regression problem, which predicts the weed bounding box and associate class probabilities without proposals generation (Redmon and Farhadi, 2017a).

A two-stage detector involves two stages as the name implies. A sparse set of candidate object boxes are first generated. Then, they are further classified and regressed (Zheng et al., 2019a). A common example of the two-stage object detector is the fast region-based convolutional neural network (Fast R-CNN) (Girshick, 2015). The Fast R-CNN is an advancement of the earliest object detector (R-CNN) (Girshick et al., 2014). The Fast R-CNN combines bounding box regression and target classification to solve multitask detection (Zheng et al., 2019a). A better two-staged object detector was proposed by the faster region-based network (Faster R-CNN) combined with the region proposal network (RPN) (Ren et al., 2015). The RPN utilises the intersection-over-union (IOU) (Nowozin, 2014) between the object proposals and the ground truth to process the input image, predict the class probability and bounding box of each crop region proposal (Zheng et al., 2019a). Two-stage detectors are not suitable for the realistic application of weed detection due to their time consumption and their inability to process large datasets. Also, after classification, post-processing is used to refine the bounding boxes, eliminate duplicate detection and re-score the boxes based on the scene (Girshick et al., 2014). These complex pipelines are slow and hard to optimize because each component must be trained separately (Redmon et al., 2016a).

On the other hand, YOLO is computationally very efficient which makes it suitable for realistic applications such as weed detection. According to (Redmon et al., 2016a), Fast YOLO is the fastest general-purpose object detector in the literature and YOLO pushes the state-of-the-art in real-time object detection. Hence the choice of YOLO as a weed detector. Although there are many versions of YOLO, this work utilises mostly versions 2 and 4 due to the availability and ease of utilisation.

To train the network, YOLO uses a loss function that penalizes incorrect predictions of bounding boxes and object classes. The loss function includes terms for the confidence score, the bounding box coordinates, and the class probabilities. The final loss is the sum of the losses for all predicted bounding boxes in the image.

3.7 ResNet-50 and YOLO Fusion

This fusion approach involves adopting the outcome of one of the layers from ResNet-50 as an input to YOLO version 2 (YOLOv2). A typical ResNet architecture is shown in Fig. 3.4. A network layer from the ResNet-50 is specified for feature extraction in YOLOv2. The ReLU (activation49relu) layer of the ResNet-50 is utilised for the feature extraction layer. This layer will now be the input of the YOLOv2. Any of the activation layers from the ResNet-50 can be used as input to the Yolov2, except the fully connected layer. The final layers of ResNet-50 consisting of the average pooling, Fully connected layer, softmax and classification layer as seen in Fig. 3.4 are truncated and merged with the YOLO layers in Fig. 3.5 to give a new fused network architecture ResNet-50/YOLOv2 for weed detection/classification which is displayed as in Fig. 3.6. The transform layer improves the stability of the network by constraining the detected object location predictions. The transform layer extracts the activations of the last convolutional layer and transforms the bounding box predictions to fall within the bounds of the ground truth (Redmon et al., 2016b). The output layer provides the refined bounding box locations of the target objects (weeds). Thus, we obtain the final fused-YOLO pipeline as in Fig. 3.7.

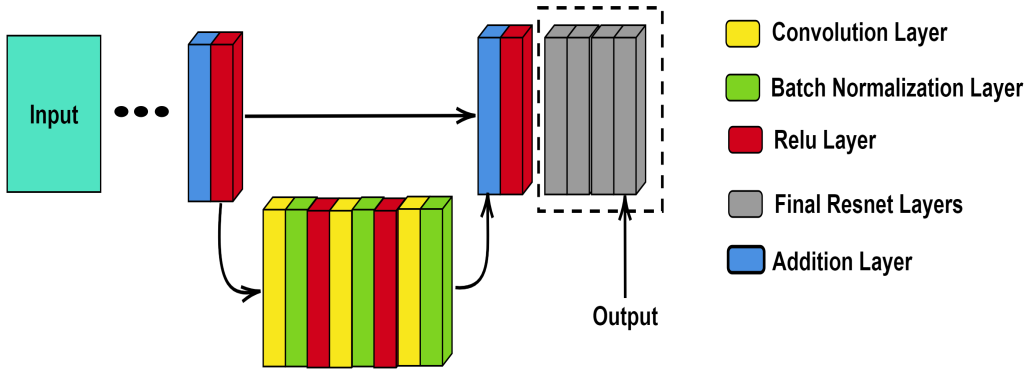


Fig. 3.4 ResNet-50 final layers architecture

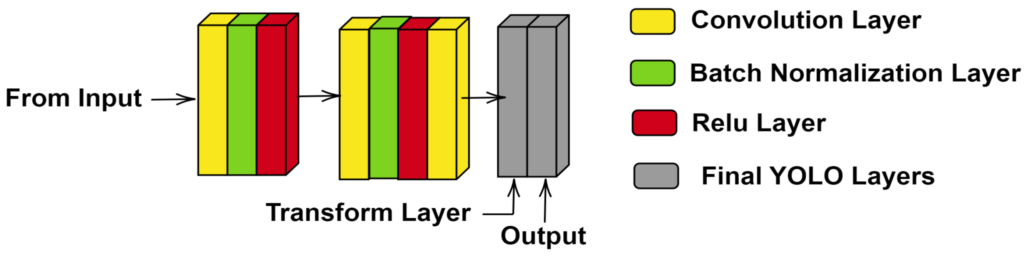


Fig. 3.5 YOLOv2 architecture

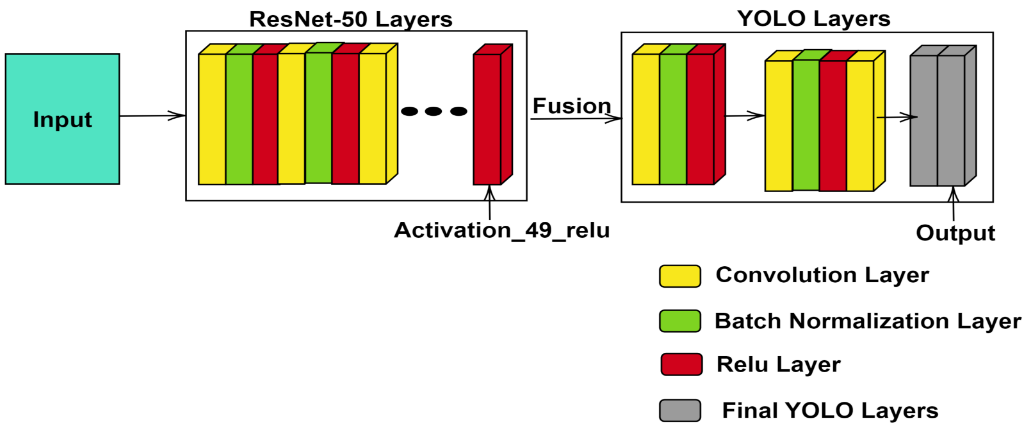


Fig. 3.6 Fusion process using a layer from Resnet as input to YOLOv2

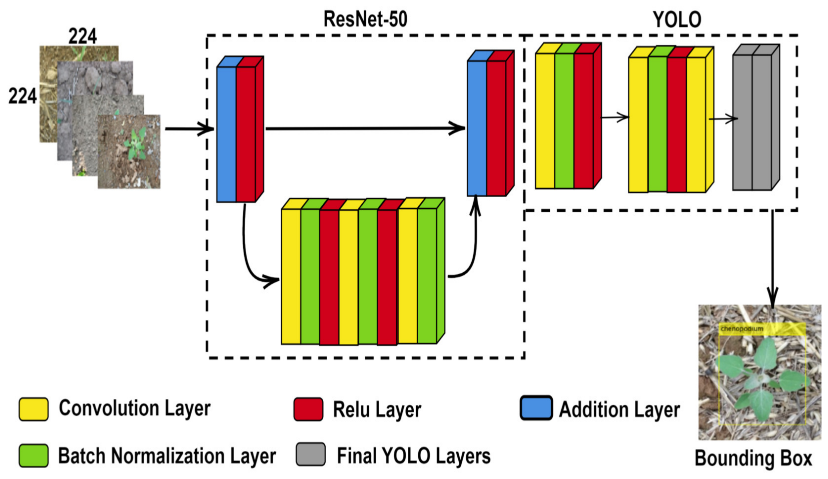


Fig. 3.7 Final fused-YOLO Deep neural network architecture

3.8 Algorithm Implementation and Testing

The main components associated with the execution of the developed algorithm can be grouped into the following:

- Bounding Box estimation
- training of the deep network
- Estimation of the training loss
- Testing of the trained network
- Estimation of the Network's precision and recall

3.8.1 Anchor Box estimation

An estimate of the anchor boxes of the labelled images is required to feed into the training phase. Anchor boxes are a set of predefined bounding boxes of a certain height and width to capture the scale and aspect ratio of the specific object class being detected. They are typically chosen based on object sizes in the training datasets. The dataset being used is a very large dataset containing up to 4000 images. There is a need of finding an optimal number of Anchor boxes that will portray the structure of the whole bounding boxes. The accuracy of this estimation is based on having a good mean IOU result. This is possible by grouping the bounding boxes with similarities through a process called clustering. In this work, the clustering method selected is called the K-medoids method. K-medoids is based on medoids (which is a point that belongs to the dataset) calculated by minimizing the absolute distance between the points and the selected centroid, rather than minimizing the square distance. As a result, it is more robust to noise and outliers than the classical K-means clustering method. K-means clustering can be achieved by initializing the cluster centroids. Each data point is assigned to the closest centroid based on the Euclidean distance between the centroid and the data point. The centroid of each cluster is updated by calculating the mean of the data points in the cluster. The process is repeated until the optimal centroid is obtained. This can easily be detected by having an unchanging centroid. This can be denoted mathematically as:

$$J = \sum_{i=1}^{k} \sum_{j=1}^{\lambda} \|\alpha_j - C_i\|^2 \tag{3.2}$$

where J is the within-cluster sum of squares, k is the number of clusters, λ is the number of data points assigned to cluster i, α_j is the j-th data point, C_i is the centroid of cluster i, and $||\cdot||$ denotes the Euclidean distance.

While K-medoids can be obtained by initializing the medoids randomly. Each data point is assigned to the closest medoid based on a distance metric (Euclidean distance or Manhattan distance). The total cost of swapping a non-medoid point is calculated for each medoid. The point with the lowest cost is assigned to be the new medoid. As in the K-means, the process is repeated until the optimal medoid is obtained. We can define it as follow:

$$J = \sum_{i=1}^{k} \sum_{j=1}^{\lambda} d(\alpha_j, M_i) \tag{3.3}$$

where J is the total distance between the data points and their medoids, k is the number of clusters, λ is the number of data points assigned to medoid i, α_j is the j-th data point, M_i is the i-th medoid, and $d(\cdot,\cdot)$ is the euclidean distance metric used to measure the dissimilarity between two points.

For this work, the K-medoid is utilised to obtain an optimum number of 7 points which in this case is the anchor boxes with a mean IOU of about 83%.

3.8.2 Network training process

As stated earlier, the network comprises of two sub-networks. The feature extraction is performed using a pretrained CNN (ResNet-50). The detection sub-network is composed of fewer CNN layers and layers specific for YOLO v2. To train the network, several inputs that parameterize a YOLO v2 network are required and they include: network input size, anchor boxes and feature extraction network. The network input size needs to be specified and typically the same size as the input images (224×224). A lower input size results to lesser training computational burden. The anchor boxes are determined using the methods in 3.8.1. Next, the feature extraction layer is selected for the training. The activation layer *activation 40 relu* is selected as the feature extraction layer. The layers after the *activation 40 relu* are replaced with the detection subnetwork. The feature extraction layer gives feature maps that are downsampled by a factor of 16 as output.

After preparing the input in form of dataset, the training of the network commences and trained over 1000 epochs with a learning rate of 0.001. A mini-batch size of 64 image samples is selected. An epoch is a full training cycle where all of the training vectors are used once to update the weights on the training set. The learning rate is a means to control the adjustment of weights in the network along the gradient.

3.8.3 Estimation of training loss

The accuracy of the training is evaluated by inspecting the training loss for each iteration. It is the Mean Squared Error (MSE) which is calculated as the summation of the localization error, the confidence loss, and the classification loss. The localization error is the measure of the error between the predicted bounding box and the ground truth box. The confidence loss measures the confidence score error when an object is detected and the objectness error when no object is detected. The classification loss measures the squared error between the class conditional probabilities for each class.

3.8.4 Testing of the trained network

From the dataset, 200 images from each class were set aside for tests. The aim is to see how correct the network is in identifying the various classes of weeds. The trained network is used for the detection/classification of weeds through all the 200 images and a table of detected bounding boxes and class labels is obtained.

3.8.5 Estimation of Network's precision and recall

Precision and recall are key factors in evaluating the overall performance of the trained network. Precision is the ability of the detector/classifier to make correct classifications while recall is the ability of the detector/classifier to find all relevant objects. The average precision is a single number that incorporates both the precision and recall of a particular class. Both Precision and recall are based on the ground truth. Mathematically, precision P is a ratio of true positive instances to all positive instances of objects in the detector as seen in Eq. 3.4 while the ratio of true positive instances to the sum of true positives and false negatives in the detector/classifier is the recall R as seen in Eq. 3.5.

$$P = \frac{T_p}{T_p + T_n} \tag{3.4}$$

$$R = \frac{T_p}{T_p + F_n} \tag{3.5}$$

where $T_p = True\ positive$, $T_n = True\ negative$ and $F_n = False\ negative$

3.9 Experimental Results

The experimental results obtained can be grouped into:

- Training loss
- Precision and recall
- Detection/Classification of weeds using testing data

3.9.1 Training loss

From Fig. 3.8, the training loss per iteration of the network is observed to assess the success of the network training phase. In total, the network went through 62,000 iterations. Sampling the first 1000 iterations for observation, we can see the training loss significantly depreciating to around 0.4 within the first 300 iterations. As more iterations are carried out, the network continues to learn more which makes the training loss keep diminishing through the next iterations. We can notice the loss maintained below 1 from the 650th iteration onwards, continuously having a maximum value of 0.8 and a minimum value below 0.2. This signifies that the network is becoming more stable in learning and hence the training loss is reducing.

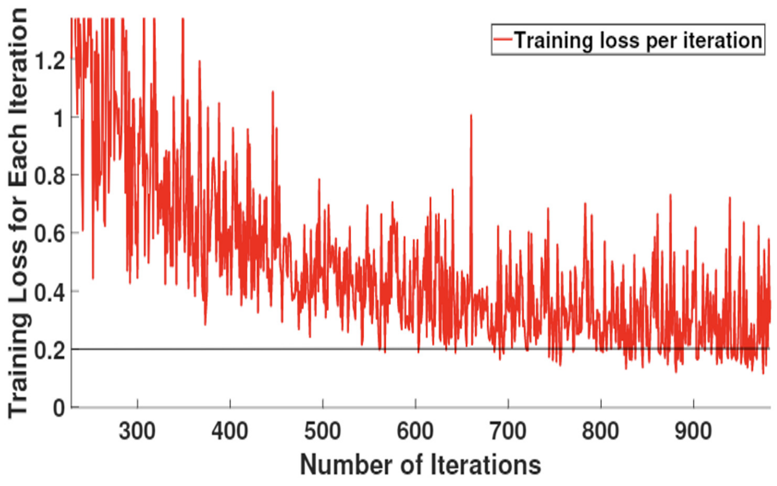


Fig. 3.8 Graph of training loss per iteration for the sampled 1000 iterations using fused ResNet- YOLO pipeline

Compared with Fig. 3.9, it can be observed that the training loss across each iteration in the fused ResNet-YOLO network is smooth and minimal whereas that of the fused faster-RCNN is rough and unstable. Even though the fused faster-RCNN started the training with quite a lesser loss as compared to the fused ResNet-YOLO, We can notice that even after the 650th iteration, the training loss still appreciates above 1.2 with some cases even up to 2 and a minimum value above 0.2. In other words, the training loss of the fused faster-RCNN is not as minimal as that of the fused ResNet-YOLO and signifies that the fused faster-RCNN learns quite slowly and have encounter more losses which in turn will affect the overall training accuracy of the network.

Thus, overall for weed classification and detection, It is worth mentioning at this point that the fused Faster-RCNN network is a resource-intensive approach with a considerable amount of time for training. The fused ResNet-YOLO on the other hand is not too demanding

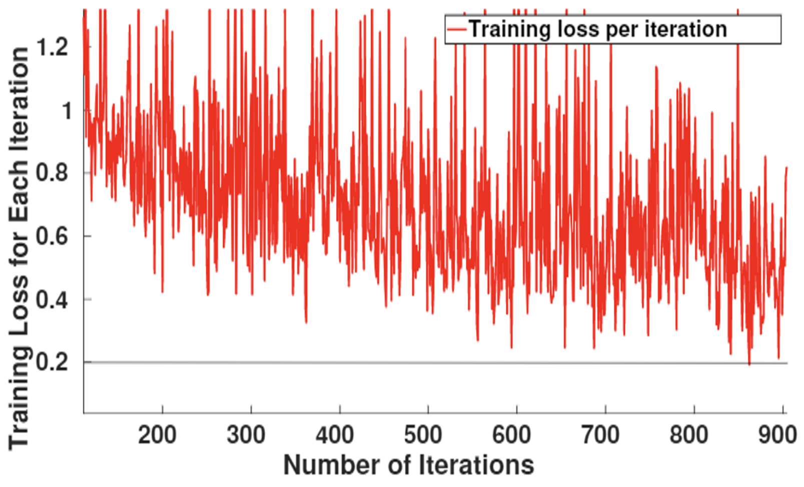


Fig. 3.9 Graph of training loss per iteration for the sampled 1000 iterations using fused Faster-RCNN pipeline

on resources and trains with much ease as compared to the fused faster-RCNN. The overall average training accuracy of the Fused ResNet-Yolo is about 99% (bluegrass 98.88%, sedge 98.52%, circium 99.11% and chenopodium 99.51%). While, the fused faster-RCNN obtained an overall training accuracy of 95% (bluegrass 98.99%, sedge 89.20%, circium 94.88% and chenopodium 99.72%).

3.9.2 Precision and recall

Fig. 3.10 shows the plot of the precision against the recall obtained from testing data. An average precision of above 93% for each weed class except the sedge weed. The decline in the precision of the sedge class is due to the dataset used in training. It was observed that the sedge class is not properly mixed i.e. there is a higher percentage of about 97% of fully grown sedge weed than infant sedge of about 3%. More so, the testing data contained a very significant number of the infant sedge weed and thus the inability of the network to detect them. Even as such, the network performed decently with about 81%. A fully stuffed dataset with the appropriate ratio of the fully grown sedge weed and the infant sedge weed will increase the precision to match that of other classes. This result depicts that the fused

ResNet-YOLO is capable of classifying and detecting weed accurately as we can see in the average precision of the individual weed classes performing very well.

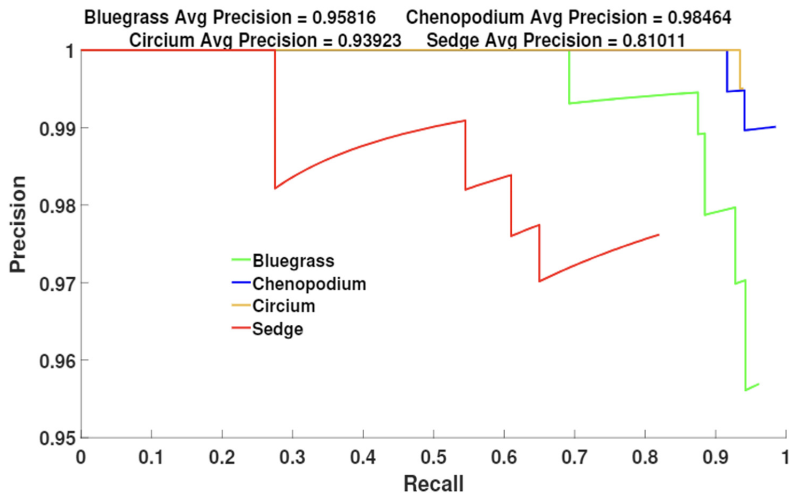


Fig. 3.10 Graph of precision and recall using fused ResNet-YOLO Network

3.9.3 Detection of weeds using testing data

A total of about 800 test images are set aside for testing. The weed detection/classifier network is used on all the images and the results obtained are summarized in table 3.1. For this analysis, we define Correct classification/detection as a situation where a class of weeds is identified correctly and a fitting bounding box is assigned. Misclassification in this case is an instance when the Network predicts a wrong class for a particular known class of weed while Misdetection is a situation where the correct class is identified as a weed but an unfitting bounding box is assigned to the weed. There are also scenarios where both conditions can occur. In Tab. 3.1, The first column shows the class of weed that is tested, the second column shows the number of correct classifications or detection of weed observed in the corresponding class and the third column shows the number of either misclassification or misdetection in the class.

The classifier/Detector was unable to classify/detect 2 weeds from the bluegrass, 32 from the sedge class. It obtained a 100% classification/detection in the Chenopodium and Circium

Table 3.1 Table showing the detections made by the fused ResNet-YOLO on the test data

Weed Class	Correct detection	Misdetection/ Misclassification
Bluegrass	199	2
Chenopodium	200	0
Circium	200	0
Sedge	168	32

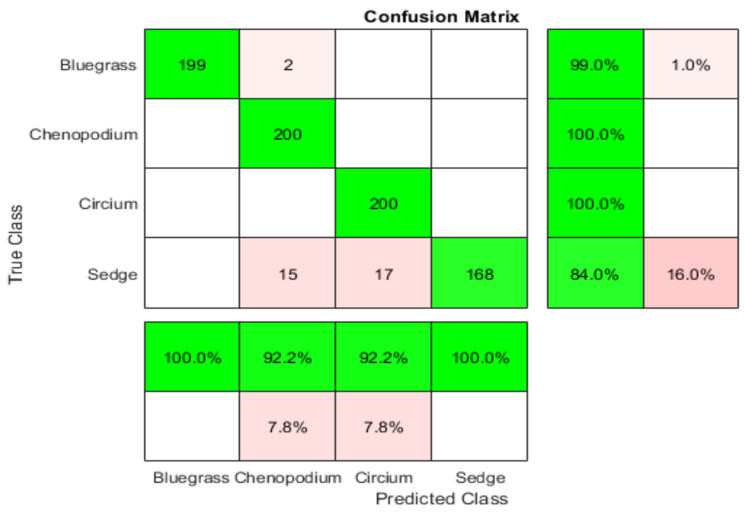


Fig. 3.11 Confusion Matrix Plot for ResNet-YOLO

class. The confusion matrix in Fig 3.11 analyses the true class versus the predicted class. It can be observed that the Bluegrass class obtained 99% correct detection and 1% misdetection. The Chenopodium and Circium classes both obtained a 100% correct detection with no misdetection from the test data. Finally, the Sedge class obtained 84% correct detection and 16% misdetection. The detection distribution percentages for the fused ResNet-YOLO are shown using a pie chart in Figs 3.12 - 3.15.

The classification/detection using the fused faster-RCNN is summarized in Tab. 3.2 using the same number of test data as in the case of the fused-YOLO. From the results, we can observe that only 62 frames were correctly detected and classified using the fused faster-RCNN model for the Bluegrass class with 138 frames from the class either misdetected

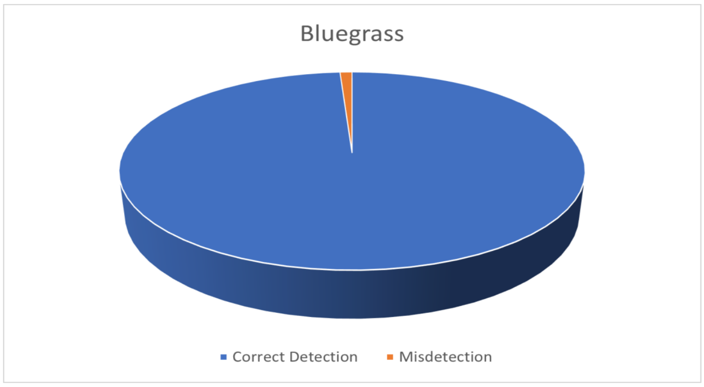


Fig. 3.12 Detection Distribution for Bluegrass Class using fused ResNet-YOLO

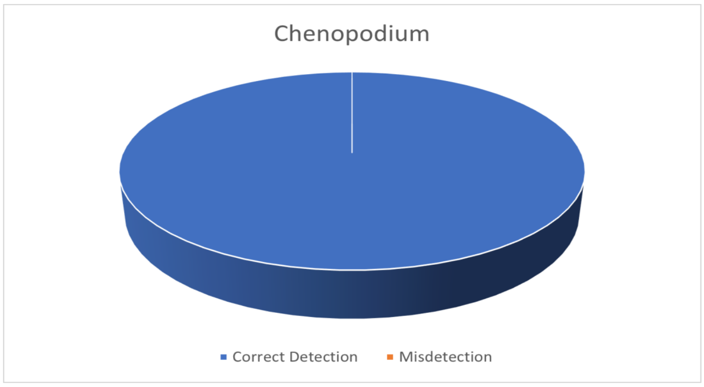


Fig. 3.13 Detection Distribution for Chenopodium Class using fused ResNet-YOLO

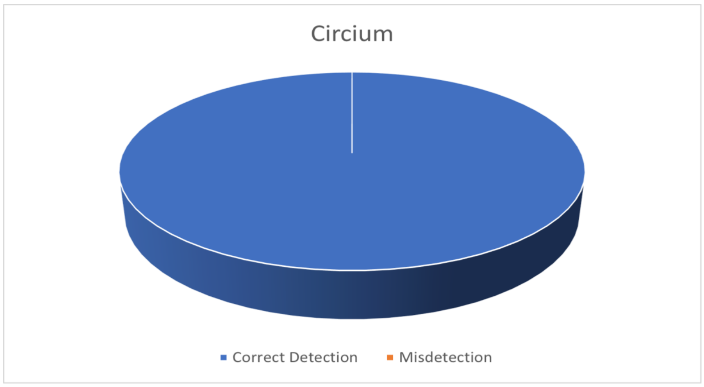


Fig. 3.14 Detection Distribution for Circium Class using fused ResNet-YOLO

or misclassified. 25 frames from the Chenopodium class and 35 frames from the Circium were correctly detected and classified. Misdetection and misclassification occurred in 175

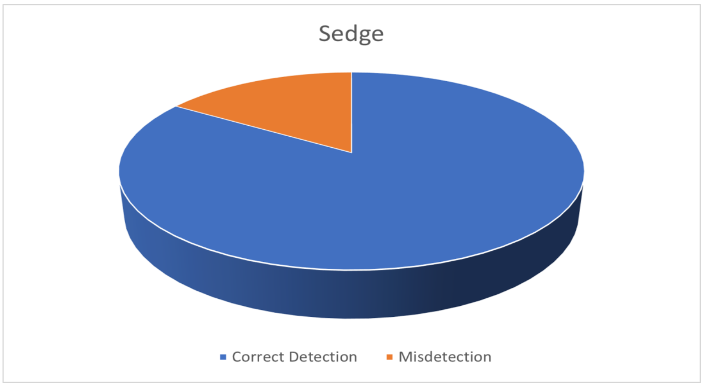


Fig. 3.15 Detection Distribution for Sedge Class using fused ResNet-YOLO

and 165 frames for Chenopodium and Circium classes respectively. Finally, the Sedge class had the lowest correct detection with only about 12 frames correctly detected and classified while 188 frames were either misdetected or misclassified.

Table 3.2 Table showing the detections made by the fused faster-RCNN on the test data

Weed Class	Correct detection	Misdetection/ Misclassification
Bluegrass	62	138
Chenopodium	25	175
Circium	35	165
Sedge	12	188

Similarly, the confusion matrix plot in Fig. 3.16 shows that the Bluegrass class obtained 31% correct detection and 69% misdetection. The Chenopodium class obtained a 12.5% correct detection and 87.5% misdetection. The Circium class obtained a 17.5% correct detection with 82.5% misdetection from the test data. Finally, the Sedge class obtained 6% correct detection and 94% misdetection. The detection distribution for the fused faster-RCNN is shown using a pie chart as shown in Figs 3.17 - 3.20.

Comparing individual class average precision from the fused faster-RCNN, we can see that the fused ResNet-YOLO network outperforms the fused faster-RCNN in all the classes. This goes to prove that our proposed fusion is more efficient for weed classification and detection.

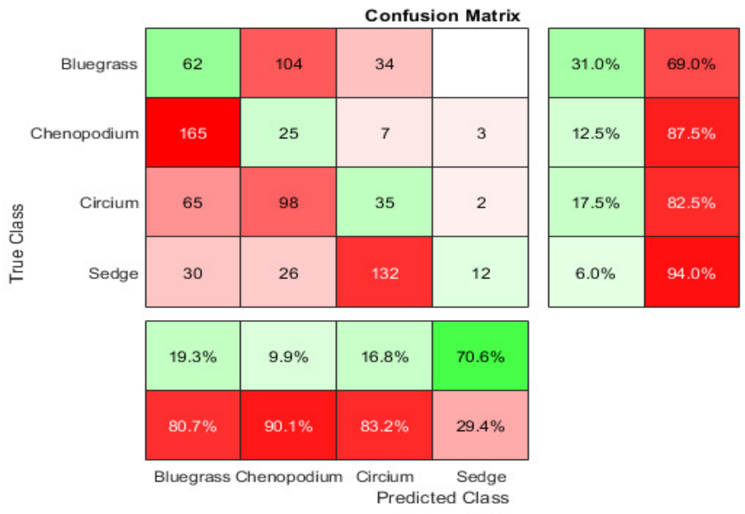


Fig. 3.16 Confusion Matrix Plot for fused faster RCNN

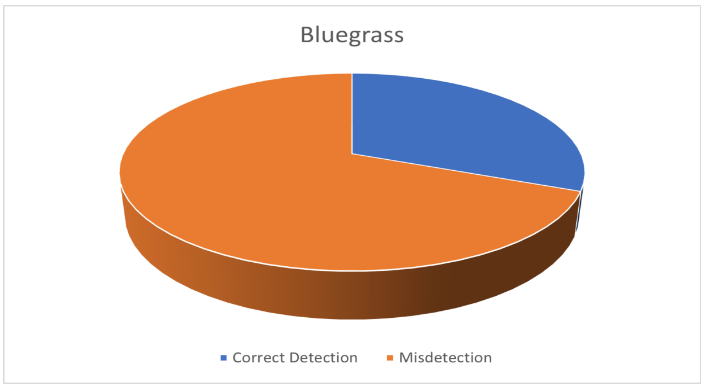


Fig. 3.17 Detection Distribution for Bluegrass Class using fused faster-RCNN

Additional images of different classes of weed (parthenium, snakeweed and prickly acacia) which contain neither of the weed classes used in training were added to test the network's accuracy and robustness. The result from this negative class in Fig. 3.21 is 100% accurate as none of the weeds out of the 30 images was identified as a weed from the trained classes.

The detection/classification samples of the four classes of weeds using the fused-YOLO pipeline are displayed in Fig. 3.22. All four weeds were detected/classified accurately with their corresponding labels(classes) displayed and bounding boxes. While Fig. 3.23 shows the

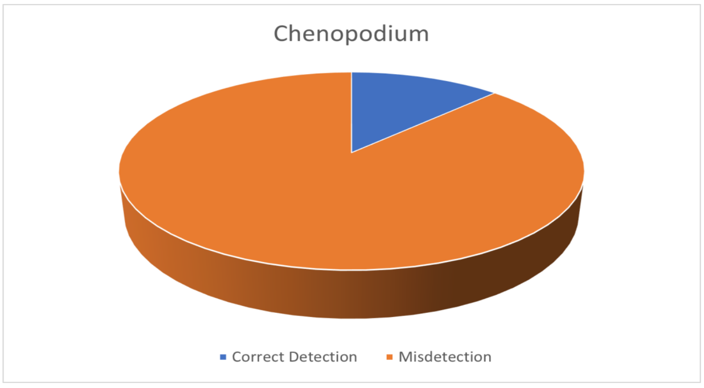


Fig. 3.18 Detection Distribution for Chenopodium Class using fused faster-RCNN

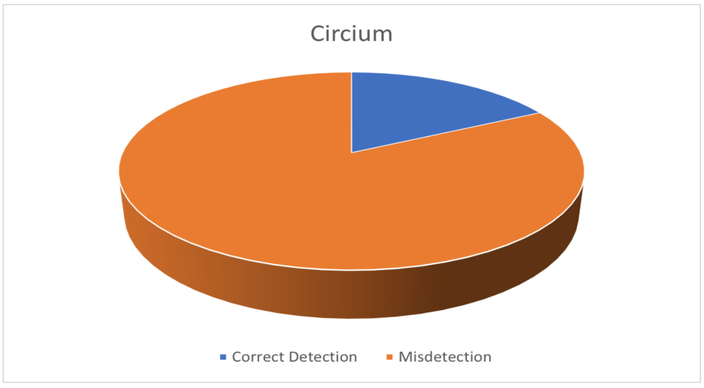


Fig. 3.19 Detection Distribution for Circium Class using fused faster-RCNN

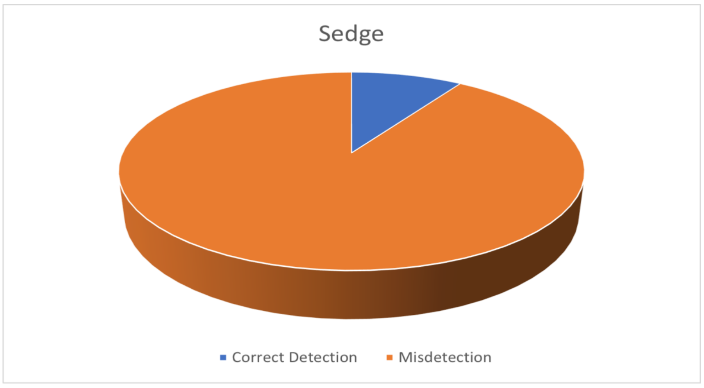


Fig. 3.20 Detection Distribution for Sedge Class using fused faster-RCNN

detection and classification using the fused faster-RCNN model. Comparing both figures, the detection using the fused faster-RCNN is observed to have some irregularities as an image

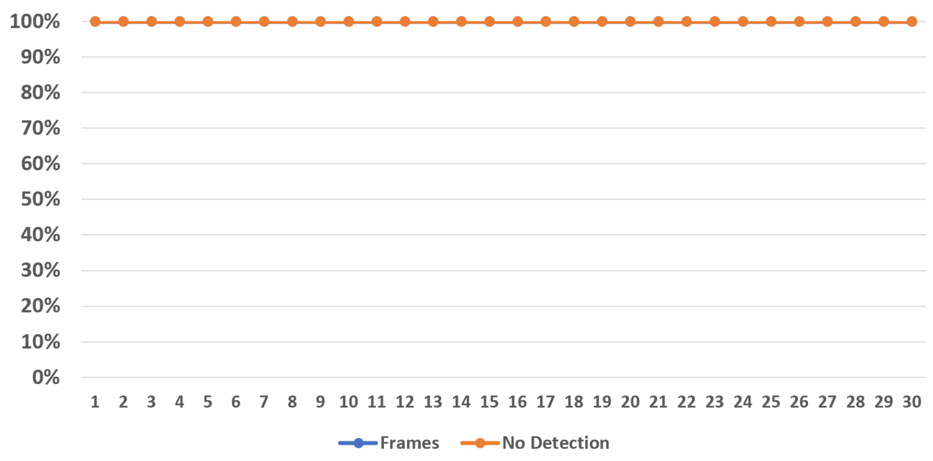


Fig. 3.21 True Negative Evaluation of the model using three different unknown classes (parthenium, snakeweed and prickly acacia) across 30 frames

with a single weed can have many false detections. This will make the approach difficult for weed detection in precision farming as the removal or spraying will not be reasonable if many bounding boxes are associated with a single target. Thus, we rely on the fused ResNet-YOLO network as the best solution for this purpose. We can mention that the detection time of the fused-YOLO is approximately 0.45 seconds while that of the fused faster-RCNN is about 1 second. If it will take a system with the fused ResNet-YOLO network an hour to cumulatively detect the weeds on a farm, the fused faster-RCNN will take about two hours as suggested by this experiment.

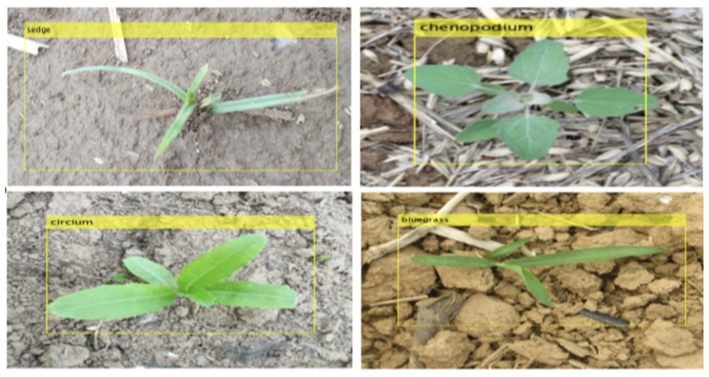


Fig. 3.22 Sample detection using fused ResNet-YOLO Network

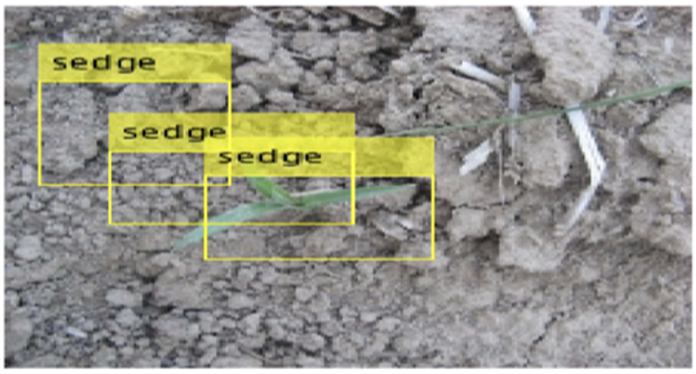


Fig. 3.23 Sample detection using fused faster-RCNN Network

3.9.4 Detection Score of weeds using testing data

The detection Score is a metric that evaluates the performance of a detection network. It measures the level of confidence that the model predicts the location and presence of the object of interest. It can be represented as a numerical value from 1-100, where a higher score indicates a more confident and accurate detection and vice-versa. Generally, we can say:

$$Score = Pr(Obj) \times IOU(bb, Obj) \tag{3.6}$$

where $Pr(Obj)$ represents the probability that an object is present in the bounding box, and $IOU(bb, Obj)$ represents the intersection over union (IOU) between the proposed bounding box and the ground truth bounding box of the object which can be expressed as:

$$IOU(bb, Obj) = \frac{Area\ of\ Intersection}{Area\ of\ Union} \tag{3.7}$$

The sigmoid function is used to map the final score to a value between 0 and 1. Furthermore, the score can be obtained as a percentage.

Figs 3.24 - 3.27 shows the detection scores of the fused ResNet-YOLO model over 200 frames. The Chenopodium and Circium class obtained scores in the range of 80 - 100. The Bluegrass and Sedge class occasionally obtained a score of 0 due to no detection in some frames.

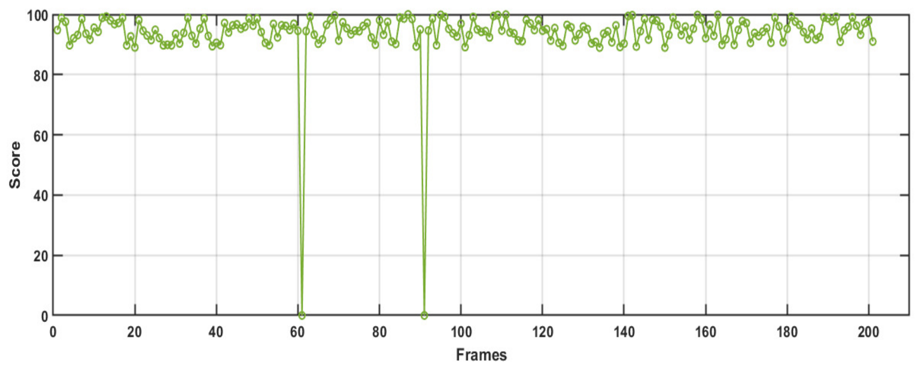


Fig. 3.24 Detection Score for Bluegrass Weed Class across Frames using the fused ResNet-YOLO

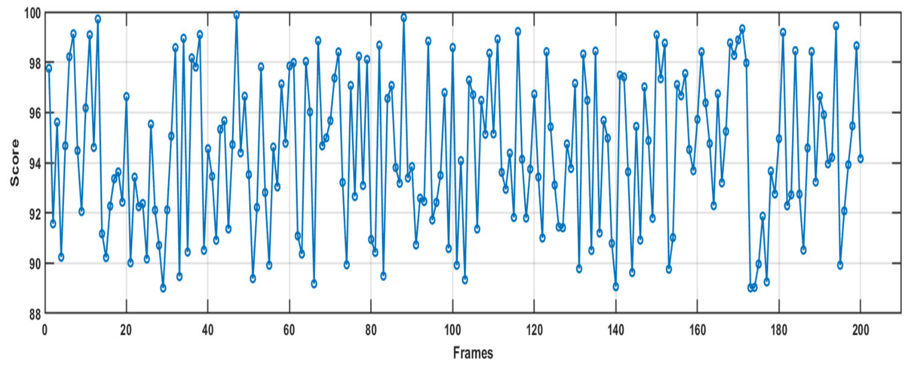


Fig. 3.25 Detection Score for Chenopodium Weed Class across Frames using the fused ResNet-YOLO

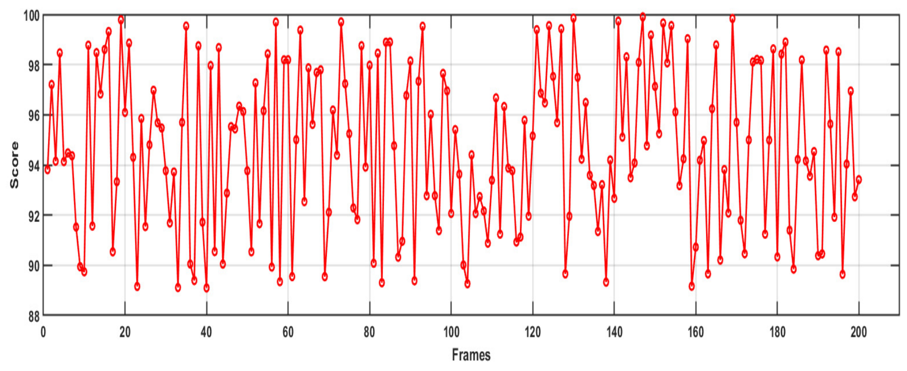


Fig. 3.26 Detection Score for Circium Weed Class across Frames using the fused ResNet-YOLO

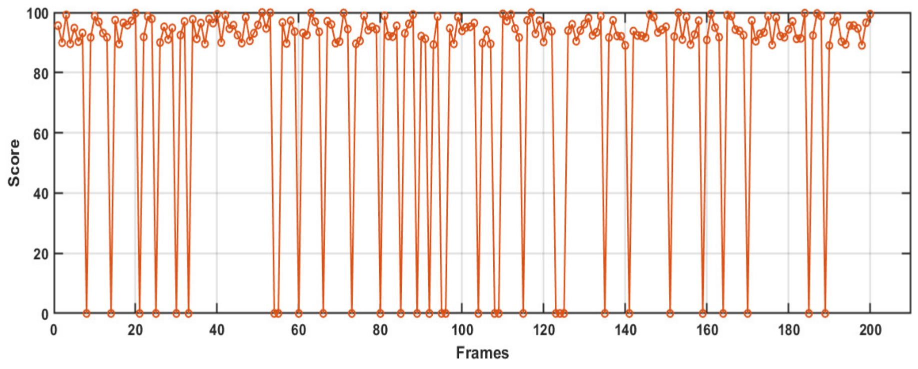


Fig. 3.27 Detection Score for Sedge Weed Class across Frames using the fused ResNet-YOLO

Figs. 3.28 - 3.31 shows the detection scores of the fused Faster-RCNN model over 200 frames. This model in most cases obtained a poor score across many frames and hence proves that it is not suitable for weed detection.

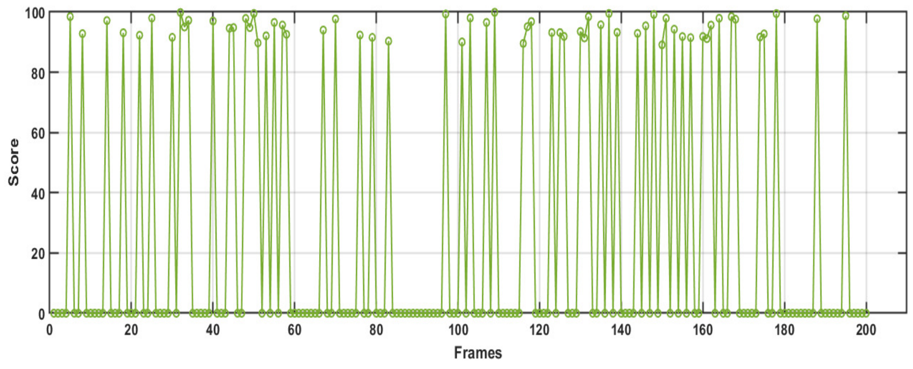


Fig. 3.28 Detection Score for Bluegrass Weed Class across Frames using the fused Faster-RCNN

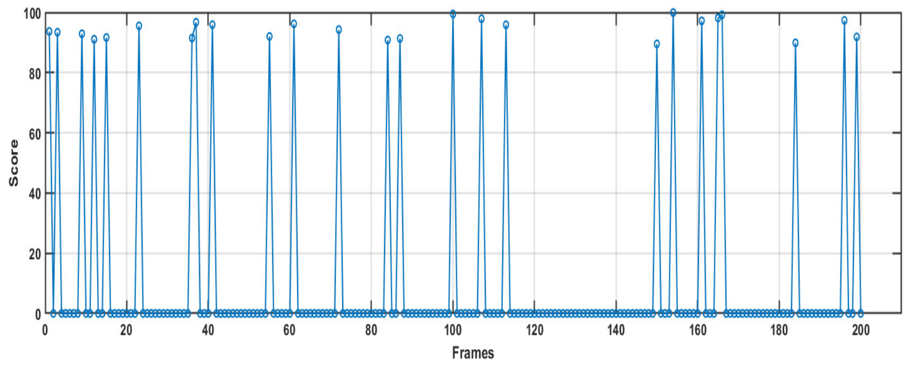


Fig. 3.29 Detection Score for Chenopodium Weed Class across Frames using the fused Faster-RCNN

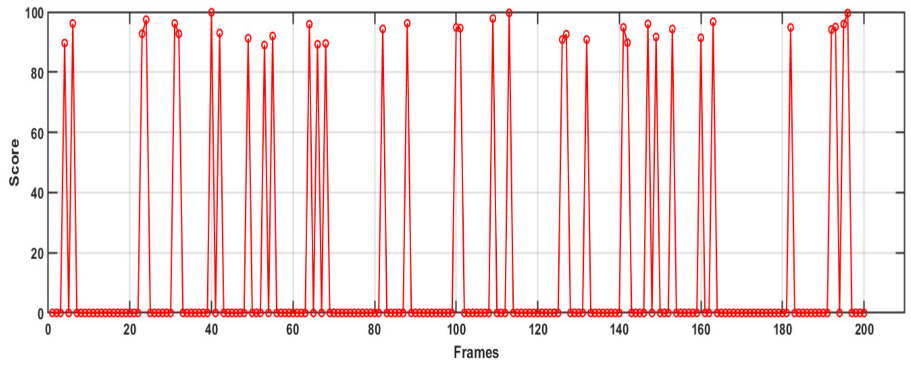


Fig. 3.30 Detection Score for Circium Weed Class across Frames using the fused Faster-RCNN

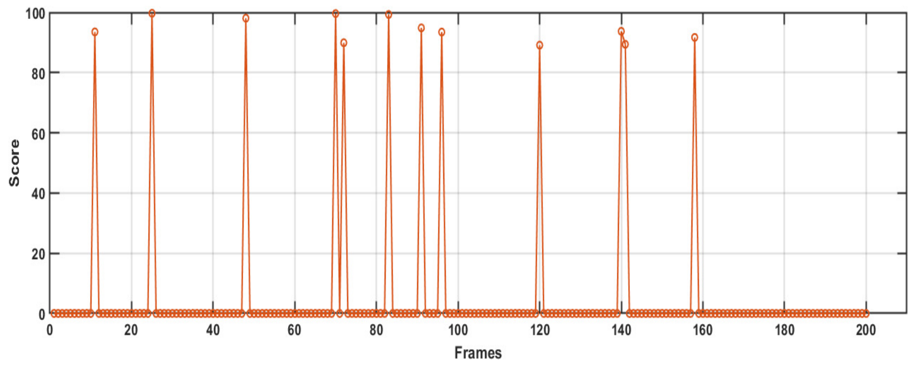


Fig. 3.31 Detection Score for Sedge Weed Class across Frames using the fused Faster-RCNN

3.10 Real-time Implementation of Network with UAV

Implementation of the proposed pipeline is performed using a cheap parrot drone with a workstation. The computation is performed on the base station equipped with intel core i7 and Nvidia GPU. ROS version 1, kinetic distribution was used to communicate between nodes. The flight controller was made on a separate node while the deep detection network was exported and integrated as a node. The real-time implementation of the trained network on the UAV can be summarised into the following:

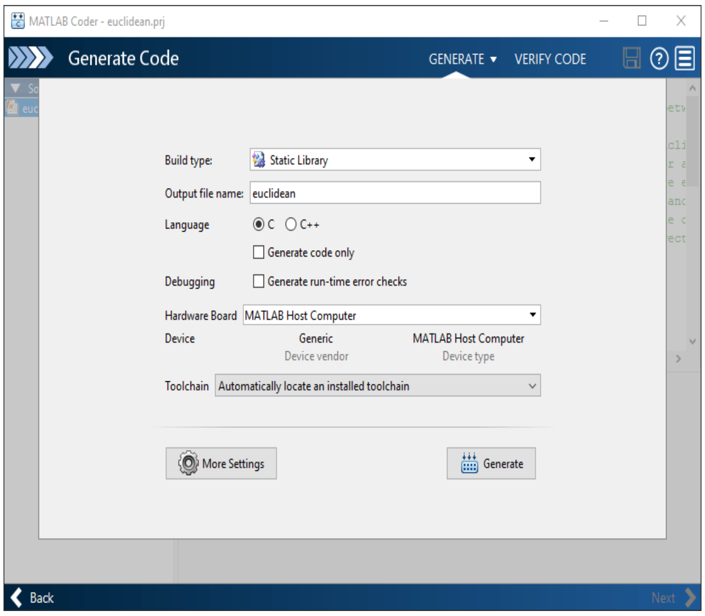
- Neural Network Exportation
- Drone Camera Calibration
- Integration of Network into ROS

3.10.1 Neural Network Exportation

The trained neural network is exported using a MATLAB GUI. Since the UAV and ROS accept a particular programming language (C++) for integration, it becomes paramount to translate the framework from MATLAB language to C++. This is possible using a MATLAB Coder interface shown in Fig. 3.32. The interface provides the user with the ability to translate, test and verify the translated packages. This can be in the form of a static library, dynamic library, or executable. The weights were translated as a static library to allow easy integration with other components. Depending on the processing capabilities the MATLAB Coder interface allows the user to select the target processor and thus compile the produced packages in the respective environment for testing before deployment.

3.10.2 Drone Camera Calibration

The Drone's camera requires calibration to correct distortion and perform proper detection as seen in Fig. 3.33. Camera calibration is the process of estimating the parameters of an image sensor (see section 2.1). The parameters which are both intrinsic and extrinsic can be


MATLAB Coder - euclidean.prj

Generate Code

GENERATE ▾ VERIFY CODE

Build type: Static Library

Output file name: euclidean

Language: [✓ C] C++

☐ Generate code only

Debugging: ☐ Generate run-time error checks

Hardware Board: MATLAB Host Computer

Device: Generic (Device vendor) MATLAB Host Computer (Device type)

Toolchain: Automatically locate an installed toolchain

More Settings Generate

Back Next

Fig. 3.32 MATLAB Coder Interface

estimated by acquiring images of the checkerboard from different perspectives. A Matlab GUIGUI is utilised to automatically estimate the camera parameters.

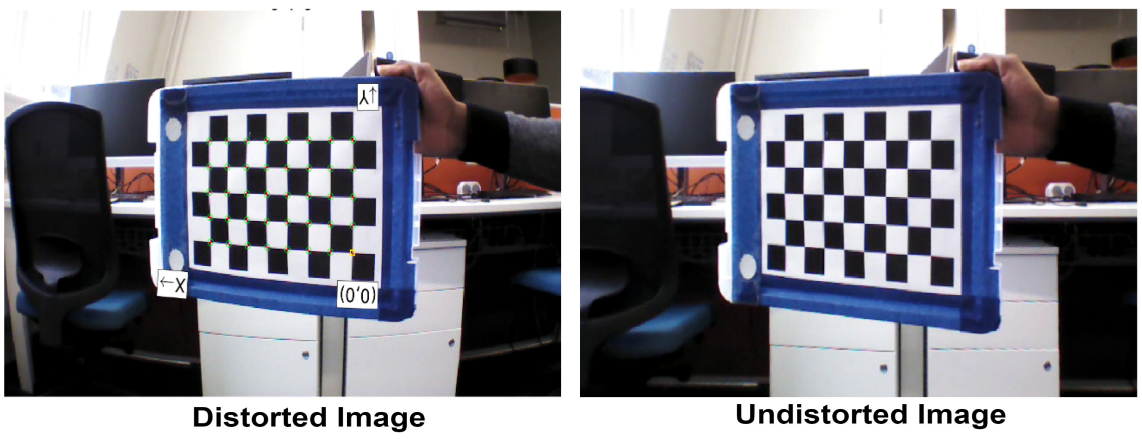


Fig. 3.33 Example of Distorted and Undistorted Images from Drone Feed

A total of 27 images of the checkerboard pattern were acquired from different angles. The dimensions of the squares of the checkerboard are known and inputted into the calibration toolbox GUI. The dimension of the square was measured to be $27mm \times 27mm$. Fig. 3.35

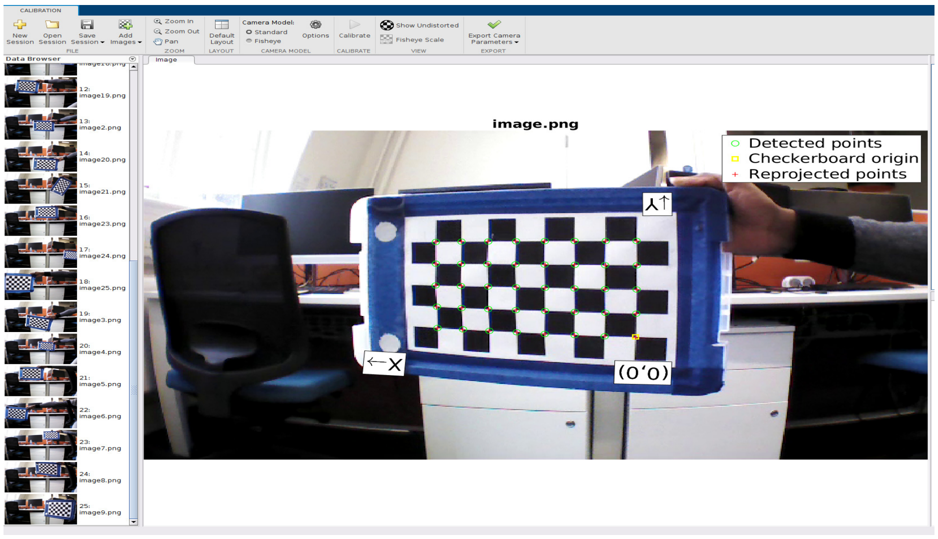


Fig. 3.34 Matlab Calibration GUI

and 3.36 show the camera-centric and pattern-centric positions of the acquisition process respectively.

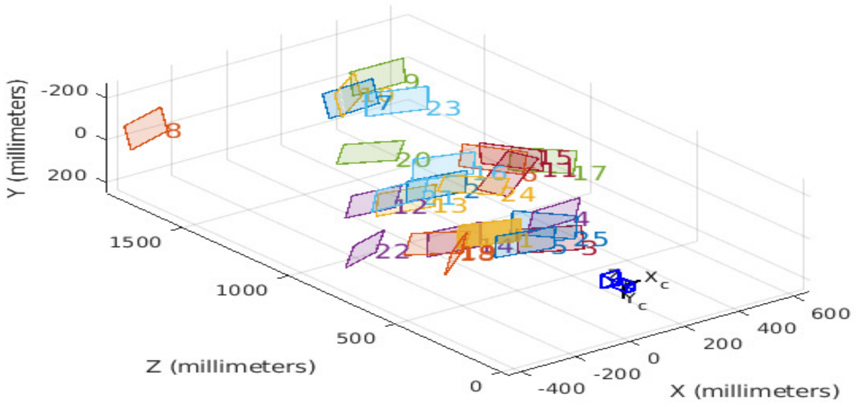


Fig. 3.35 Camera Centric Positions

The calibration obtained an overall mean error of 0.13 pixels which is a good accuracy (see Fig. 3.37). It essentially means that the reprojected pattern deviates from the original pattern by 0.13 pixels which we can conclude as having a sub-pixel accuracy.

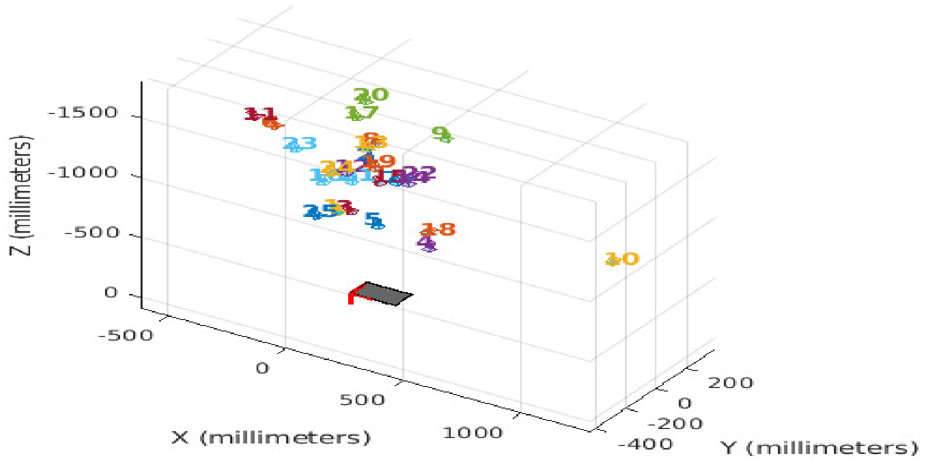


Fig. 3.36 Pattern Centric Positions

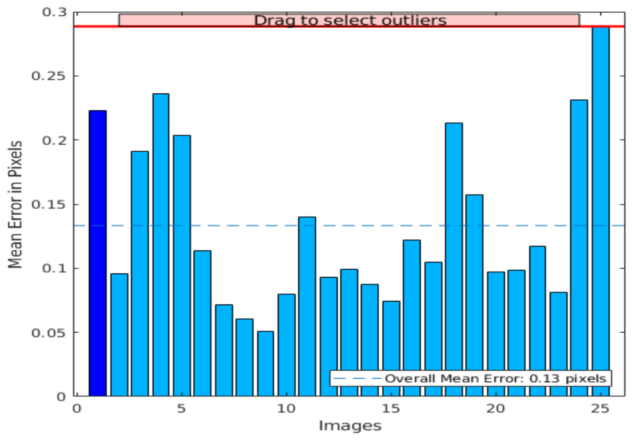


Fig. 3.37 Reprojection Error of Camera Calibration

Finally, the camera parameters (both intrinsic and extrinsic parameters) are obtained and used to rectify the images from the drone for effective detection using the trained model.

3.10.3 Integration of Fused-YOLO with ROS

The ROS interface allows users to integrate the network, control the drone and utilise the sensors on the UAV as seen in Fig. 3.38.

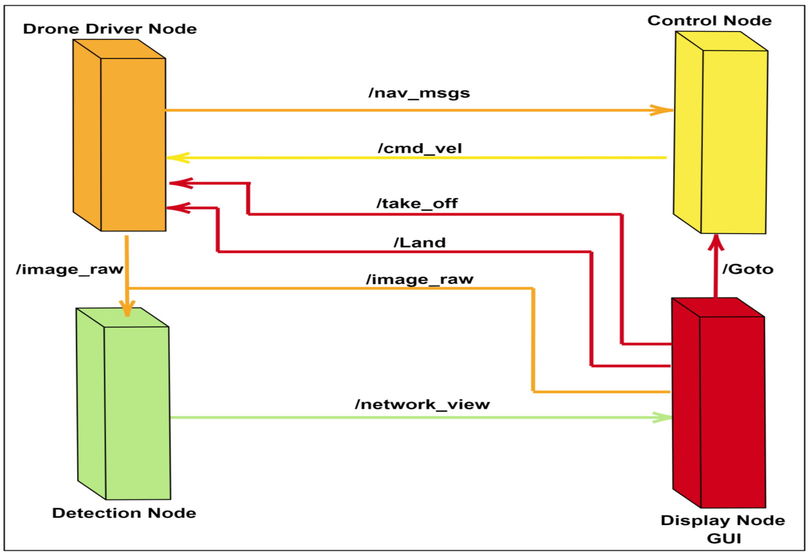


Fig. 3.38 ROS architecture Layout for Real time Test

The flight controller is made as a separate node while the deep detection network is made on another. Images from the drone's camera are obtained as a result of subscription to the camera topic. Such images are fed as input to the deep network in the form of messages to process and detect the weeds. The *cmd_vel* is the message that controls the velocity of the drone and the *nav_msg* gives information about the current state of the drone. The Land and *take_off* messages perform the landing and taking off of the drone respectively. The *image_raw* message is the image from the drone's camera. The network view message is the output of the detector network which is an image with the detected weed and bounding box. The driver node is the drone itself, the control node controls the flight of the drone, the detection node encompasses the deep network model for detection, the display node displays the output of the detection network and the GUI is a graphical user interface for issuing basic commands. Thus, the ROS layout can be subdivided into the following nodes:

- Drone Driver Node
- Control Node
- Detection Node

- Display/GUI Node

Drone Driver Node

The drone driver node provides a high-level interface between the sensors on the drone and the ROS software. All the information from the sensors such as camera images, rotor speed, altitude, odometry, battery level, IMU data, etc. can be accessed from this node. This allows the user to access these messages in the form of a subscription, process the messages and manipulate the messages in the form of publication to implement a desired outcome. The Node allows users to implement tasks such as flight control, obstacle avoidance, path planning, object detection and other algorithms that are based on sensor data. The drone driver is available as open source (Monajjemi, 2014).

Control Node

The control node is important as it controls the flight of the drone. It takes in a *goto* message from the GUI node which is essentially position coordinates relative to the current position of the drone. The control node sends *cmd_vel* which are velocity messages to the drone driver and receive a *nav_msgs* which is the current state of the drone as feedback to adjust the controller accordingly. The drone is then able to move to the desired position. This node is obtained open source as part of the *tum ardrone* package (Lesire-Cabaniols, 2014).

Detection Node

The detection node houses the deep network model. The node utilises the exported neural network library to detect and classify weeds. Image frames in the form of *image_raw* messages are received from the drone driver. The frames are used as input to the detection network to detect and classify the object of interest by the network. The node gives an output of frames with detected objects in the form of *network_view* messages. The summary of the process of the detection node is shown in Fig. 3.39

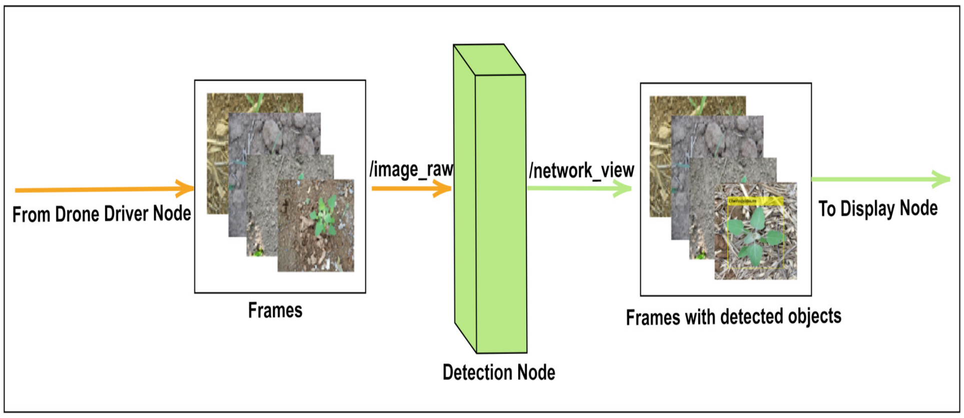
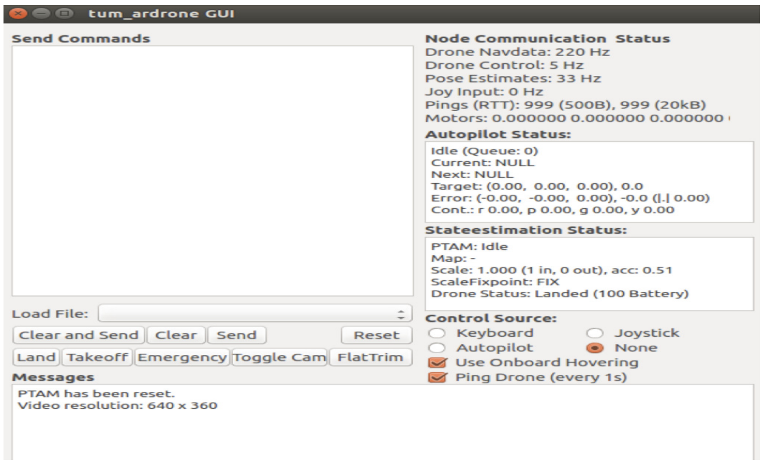


Fig. 3.39 Input-Output of Detection Node

Display/GUI Node

The display/GUI node has two components. The display component and the GUI component. The display component shows the output frames from the detection network. It receives frames in the form of *network_view* messages and then displays the frames so users can see the detected object in real-time. The GUI component is a graphical interface that allows users to execute commands in the form of messages. Messages such as taking off the drone *take_off*, landing the drone *land* and moving/controlling the drone *goto* are published from this node. Fig. 3.40 shows the graphical interface used in commanding the drone.



PTAM has been reset.
Video resolution: 640 x 360

Node Communication Status
Drone Navdata: 220 Hz
Drone Control: 5 Hz
Pose Estimates: 33 Hz
Joy Input: 0 Hz
Pings (RTT): 999 (500B), 999 (20kB)
Motors: 0.000000 0.000000 0.000000

Autopilot Status:
Idle (Queue: 0)
Current: NULL
Next: NULL
Target: (0.00, 0.00, 0.00), 0.0
Error: (-0.00, -0.00, 0.00), -0.0 (|.| 0.00)
Cont.: r 0.00, p 0.00, g 0.00, y 0.00

Stateestimation Status:
PTAM: Idle
Map: -
Scale: 1.000 (1 in, 0 out), acc: 0.51
ScaleFixpoint: FIX
Drone Status: Landed (100 Battery)

Control Source:
Keyboard
Joystick
Autopilot
[✓ None]
[✓ Use Onboard Hovering]
[✓ Ping Drone (every 1s)]

Fig. 3.40 GUI Node for commanding drone

3.11 Conclusions and Future Work

This chapter aimed at contributing to innovative deep-learning technology for precision farming by identifying, detecting, and classifying the various available weeds. From the experimental results obtained, It can be justified that this approach will be 99% effective in classifying weeds. Not limited to classification, this method can identify the ROI (weed) by drawing a bounding box around it with the associated label displaying the specie of weed. As precision agriculture involves precise inputs for precise outputs, the pipeline can reduce the excessive use of harmful chemicals that damages the ecosystem on weed treatment by precisely detecting and isolating the target from the background. This will facilitate the application of UAVs and ground robots in weed detection and selective spraying/weeding and hence contribute to precision agriculture and environmental sustainability. The pipeline has been implemented in real-time using a parrot drone to detect weeds. The network is light enough on the processor and did not suffer from significant latency during the experiments. For future work, the accuracy of the pipeline can be improved while optimising the computational burden. Also, a larger dataset containing more similar-looking species can be acquired to further evaluate the robustness of the network.

Chapter 4

Relative Position Estimation of Detected Weeds

Chapter abstract

This chapter presents a novel monocular vision-based approach for drones to detect multiple types of weeds and estimate their positions autonomously for precision agriculture applications. Building on the previous Chapter, the success of the proposed model for weed classification and detection in Chapter 3 inspired the model's extension to relative pose estimation. Images are acquired from a monocular camera mounted on the UAV following a predefined elliptical trajectory. The detection/classification network earlier proposed is complemented by a new estimation scheme adopting an Unscented Kalman Filter (UKF) to estimate the exact location of the weeds. Bounding boxes are assigned to the detected targets (weeds) such that the centre pixels of the bounding box will represent the centre of the target. The centre pixels are extracted and converted into world coordinates forming azimuth and elevation angles from the target to the UAV, and the proposed estimation scheme is used to extract the positions of the weeds.

4.1 Motivation

Re-iterating the importance of weed control, the productivity of agriculture is threatened by the existence of weeds which are parasitic (Pusphavalli and Chandraleka, 2016b). Weeds are unwanted plants that grow on the farmland and compete with the desired plants for water, nutrients, space and sunlight. The losses in productivity reach 25% in Europe, but in the less developed areas in Africa and Asia, almost half of the potential food yield is lost due to weeds (Altieri et al., 1977). It is reported that lettuce yield is reduced over 50% due to weeds (Lanini et al., 1991), wheat yield is reduced by 15% (Hodgson, 1968) and there is up to 71% drop in seeded tomato yield (Monaco et al., 1981) due to weed infestation.

Conventionally, weeds are removed using crude tools and herbicides. However, these processes are wasteful and dangerous to the environment since these herbicides are made of harmful chemicals as earlier established. To efficiently remove weeds, there is a need to carefully identify the weed, then localise its exact position, then finally apply the right quantity of herbicides or deploy the appropriate tool for the weed removal. This gives rise to the consideration of robotics platforms for weed removal. While some works have proposed mechanical robotic tools (Bakker et al., 2006; Pusphavalli and Chandraleka, 2016b), others proposed robotic sprayers to reach the objective (Gonzalez-de Soto et al., 2016; Malneršič et al., 2016; Oberti et al., 2016; Tang et al., 2016).

Weeds can be identified using computer vision techniques (Herrera et al., 2014; Pusphavalli and Chandraleka, 2016b) and the adoption of deep neural networks for weed detection is increasing (Lottes et al., 2017b; Yu et al., 2019b; Zheng et al., 2019b). To bridge the gap between weed detection and precise weed removal, there is a need to address the problem of identification and localisation of weeds.

Commercially available systems for smart weed detection and removal are generally expensive. The availability of affordable off-the-shelf UAVs with essential sensors makes it pertinent to exploit them for this research. The fusion of the information from several sensors through a Robotics Operating System (ROS) (Quigley et al., 2009) framework makes it possible to perform several complicated tasks through effective communication between the sensors. Works such as navigation (Guimarães et al., 2016), path planning (Marin-Plaza et al., 2018) and localization of targets (Ruiz and Aouf, 2017) have been possible through this setup. Thus, this can be extended to exploit an affordable platform for localizing and estimating the relative position of weeds.

Recall from Chapter 3 that the proposed model is effective in efficient weed classification and detection, this success drives the need to elevate the work to the next step which is localisation. Localisation of the detected weed is important to facilitate the removal or spraying of the weed. The process intelligently gives the robot an idea of the approximate location where targeted weeds are situated.

A parrot drone platform is subjected to a predefined elliptic trajectory, and the stream of images from a monocular camera mounted on it is acquired throughout its motion. The stream of images is utilized to precisely detect the object of interest in the images using the fused-YOLO network which is a cascaded ResNet-50 (He et al., 2016a) and YOLOv2 (Bochkovskiy et al., 2020). The detected weeds are then assigned to bounding boxes and the centre pixels of the assigned bounding boxes are extracted. The centre pixel is assumed to be the centre of the detected weed, and it is transformed from the image frame to world coordinates. Azimuth and elevation angles of the target centre point with respect to the UAV are extracted and later fused in the Unscented Kalman Filter (UKF) (Wan and Van Der Merwe, 2000) to estimate the location of the weed. The contributions of this method are twofold:

- Utilizing an affordable platform equipped with a monocular camera for accurate multiple target position estimation.

- The extension of a Deep Neural Network (DNN) output beyond detection and identification but to novel relative position estimation such that, the information obtained from the detection network (bounding boxes, ROI pixels) can be further processed to achieve relative position estimation of the detected target.

4.2 Related Work

Weed detection using feature extraction in image processing is the earliest technique used to identify weeds with computer vision. Edge detection has been utilized as a technique for weed detection (Gomez-Balderas et al., 2013; Paikekari et al., 2016). However, the main plant and weed can not be effectively differentiated using edge detection only. Different illumination conditions can be used to improve the detection using a colour model and split component of grey images (Tang et al., 2016). A vertical projection method and a linear scanning method are combined to quickly identify the centre line of the crop rows. However, in this method, it is assumed that every plant detected outside the centre line of the crop rows is a weed. This is not always the case as weeds can also grow along the center line. Machine learning techniques provided results that perform better if weeds are not on the centre line of the crop rows (Islam et al., 2021; Lottes et al., 2017b). Nevertheless, all these methods are not capable of precisely detecting the exact specie of weed. These limitations prompted the use of deep neural networks for weed detection and classification.

Weed detection was performed for perennial rye-grass with deep learning convolutional neural network (Yu et al., 2019b). The work concluded that VGGNet (Simonyan and Zisserman, 2014b) performed better with the rye-grass dataset. This performance can be improved by capturing sequential information and combining RGB and Near Infrared (NIR) images (Lottes et al., 2018b). The drawback to this work is again the lack of weed species classification for accurate herbicide selection. Another work investigated the combination of classification and detection for fruits (Zheng et al., 2019b). Similarly, in the previous chapter, we combined a classification and detection network for weed detection. This way, we can categorically tell the type of weed and identify a Region of Interest (ROI) for further processing.

The accuracy of weed detection can be impacted by many factors such as variable lighting conditions, sun angles, occluded and damaged plant leaves and changing morphological or spectral properties of plant leaves at different growth stages (Liu and Bruch, 2020). It becomes imperative to use a rich dataset for training with different conditions. The conventional four steps in the procedure for using ground-based machine vision and imaging processing techniques in weed detection are pre-processing, segmentation, feature extraction and classification (Wang et al., 2019a). We aim at extending this procedure to localising and estimating the relative position of the weed.

The crowded literature on target localisation can be grouped according to the platform used (Hou and Yu, 2014; Redding et al., 2006), or the sensors employed (Deneault et al., 2008; Ruiz and Aouf, 2017) or the estimation model studied (Hosseinpoor et al., 2016; Redding et al., 2006). The main aim for all is to maximize localisation accuracy and minimize the time required. Few however seek to use small UAVs with affordable sensors to achieve high performance. A combination of a 2D laser range finder with a monocular camera can be used for the localisation (Hou and Yu, 2014). Although the maximum deviation recorded using this method was about 13% from the actual measurement. This may be due to the not-so-robust target detection process employed in the work. Edge detection and colour detection were utilized to detect the only green circular target in the scene. In reality, there can be many targets with seemingly similar features. Moreover, using this approach to detect the target while in flight can cause target blurring. An alternative approach was presented based on real-time kinematic positioning and thermal imagery (Hosseinpoor et al., 2016). This approach is based on the assumption that as long as a ground rover and a base station maintain at least 5 satellites in common, there can be an accurate prediction of the rover's location.

The first to exploit the combination of UAV state estimates with the image data to acquire bearing measurements of the target and utilize them in the target localization is (Ponda et al., 2009). In their work, a fixed-wing UAV was subjected to numerous trajectories to find the optimal trajectory for target localization. The image data of the target are processed to obtain bearing angles from the drone to the target, and an extended Kalman filter is used for position estimation. Even though it is a simulation work, good estimation results were obtained after 50 measurements for a single target. Another work also attempted the estimation with a fixed-wing UAV and using a Recursive Least Squares filter but suffered a wide error of 10.9m (Redding et al., 2006). To tackle the limitations of fixed-wing UAVs, particularly in manoeuvrability and altitude of flight, a quad-rotor can be utilised (Redding et al., 2006). Accurate results were obtained after 30 seconds for a single target using this method.

4.3 Method for Relative Position Estimation of Detected Weeds

The methodology we propose for relative position estimation of detected weeds can be summarised as follows:

4.3.1 Problem Definition

To effectively provide a befitting solution for weed position estimation, the overall problem has to be discussed. The problem is to estimate the exact position of the weed using a UAV with no sophisticated sensors. An affordable platform equipped with a monocular camera with no sufficient information such as depth being generated makes it difficult to estimate the positions of weeds relative to the platform. The idea is to utilize the camera to detect a target and utilise the detection bounding boxes to estimate the target's position. To do so, first, the platform identifies/detects and localises the target in the image frame. The trained network is used for the target detection and some post-processing is performed to localise the target in the image frame. Secondly, the information from bounding boxes is used to estimate the centre position of the target in the image frame. Since the objective is to solve with a monocular camera set-up, where the depth information is not readily at hand as in the case of a stereo camera set-up, the bounding box's centre pixels are converted to bearing angles and afterwards into azimuth and elevation angles (further explained in section 4.3.2) with respect to the UAV. Thus, the problem can be divided into

- Acquiring the images from a monocular camera and transferring them to the ground station together with the position of the UAV.
- Detection and classification of the weed from a monocular camera.

- Extracting the position of the weed in the image frame.
- Estimating the position of the weed in the world frame.

To have an accurate estimate of the world coordinates of the weed, measurements regarding this information should be rich. The UAV is controlled to make a predefined ellipse trajectory. The nature of the trajectory is an important factor in the estimation accuracy as the field of view (FOV) varies from one point to another along the trajectory. Since the targets are at a stationary position, the trajectory choice will be such that the FOV of the camera can be limited to cover all the targets at each point along the trajectory so that updates can be obtained from each target simultaneously. A constant trajectory altitude of $1m$ is selected. The bearing angle measurements for the position of the weed are fused in a UKF framework.

4.3.2 Technical and Theoretical Approach

The solution is summarised in a process flow chart shown in Fig. 4.1. The process flow encompasses mainly the data acquisition section, the ROS nodes on the ground station and the output section. The input data acquisition section is the hardware that provides inputs to the system. The image stream from the monocular camera mounted on the drone, the ground truth positions of the UAV and the target (weed) measured by the tracking system are the inputs to the process. The ROS nodes hosted on a ground station with core-i7 processing power are the main processing part of the system. Detection and classification of the weed using a Deep Neural Network (Fused-YOLO), centre pixel extraction on the images, calculating the bearing angles, fusing the bearing angles to estimate world coordinates of the weed with UKF, and trajectory planning with drone driving tasks are performed on the ROS nodes. The output section consists of the position coordinates of the estimated target (weed).

Unmanned Aerial Vehicle (UAV) and Tracking System

The UAV used is an affordable off-the-shelf Parrot AR drone. It is a six-degree-of-freedom quadcopter with a miniaturised IMU, an ultrasonic sensor, a frontal camera with 720p sensor and 93° lens, a downward/vertical camera with QVGA sensor with 64° lens, and 4 brushless

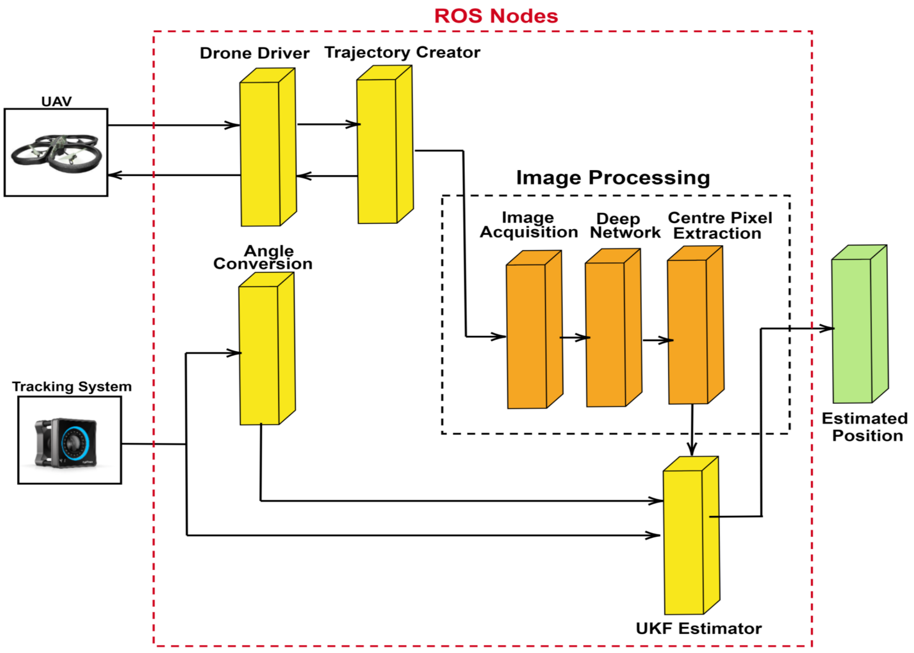


Fig. 4.1 Process Flow of the Relative Position Estimation Pipeline

14.5-watt, 28.500 RPM in-runner type motors (Parrot, 2020). The tracking system is a set of cameras with 1.3 MP resolution, +/- 0.30 mm 3D accuracy, 240 FPS native frame rate and 1000 FPS max frame rate which are used for tracking (Motive, 2020).

Drone Driver

The drone driver is a ROS package that consists of all the libraries of the parrot drone's sensors and inbuilt controllers. The drone driver is utilised to control the drone and also to receive image feeds from the drone's camera. The planar velocity references in UAV frame $V_x^{(u)}$ and $V_y^{(u)}$ indicated with the superscript u are transferred from the reference position derivatives defined in the ground frame indicated with the superscript g, namely $\dot{X}_d^{(g)}$ and $\dot{Y}_d^{(g)}$ in this drone driver ROS node as well:

$$(V_x^{(u)}(t), V_y^{(u)}(t)) = T_g^u(\dot{X}_d^{(g)}(t), \dot{Y}_d^{(g)}(t)) \tag{4.1}$$

where T_g^u is the reference frame transformation from ground frame to world frame.

Trajectory Creator

This node provides the profile of the trajectory to be performed by the drone and updates the drone driver with the necessary control parameters to follow this trajectory. An ellipse trajectory is employed taking inspiration from the circular trajectory proposed for target localisation (Gonzalez-de Soto et al., 2016). This is modified in this work to an ellipse so that the neighbouring targets can fit into the field of view (FOV). The trajectory profile is defined as follows:

$$\begin{aligned} X_d^{(g)}(t) &= a\cos(\omega t) \\ Y_d^{(g)}(t) &= b\sin(\omega t) \end{aligned} \tag{4.2}$$

Therefore,

$$\begin{aligned} \dot{X}_d^{(g)}(t) &= -a\omega\sin(\omega t) \\ \dot{Y}_d^{(g)}(t) &= b\omega\cos(\omega t) \end{aligned} \tag{4.3}$$

where a and b are radii in x and y axis, respectively. ω is the angular velocity and t is time.

Image Acquisition

This node receives image data from the drone driver and distributes it to the detection network via an image transport link. Images are transported in the form of messages at a frequency of up to 40Hz so they can effectively be utilised by the detection network. The drone's camera was calibrated beforehand and the parameters were obtained as explained in section 2.1.

Deep Network

Conventionally, approaches such as colour detection or edge detection are deployed for detection and localisation problems (Mueggler et al., 2014; Ruiz and Aouf, 2017). However, most often weeds have about 90% resemblance with the main plant. Taking this into consideration, the previously proposed fused network in Chapter 3 is incorporated with extensions to suit this peculiar problem. Recall that, the network is a cascade of a classification network ResNet-50 and a detection network YOLO. A detection network is necessary since using a classification network alone will classify the entire image as a weed which includes the region of interest and the background without categorically indicating the weed within the image. The interest mainly lies with the bounding boxes in this case. So that the region of interest within the image corresponds to the weed. The choice of the network is pertinent to the accuracy obtained in fruit classification (Zheng et al., 2019b) and speed in weed detection (Abdulsalam and Aouf, 2020). As earlier established, the architecture is 95-98% effective in weed classification and detection. The network is trained with a dataset of 2000 images of the weed. The deep neural network and its extension used for this work are shown in Fig. 3.7.

The input layers of the trained network are not compatible with the output coming from the drone camera. An encoding-decoding operation is performed as shown in Fig. 4.2 to remap and rearrange the pixels.

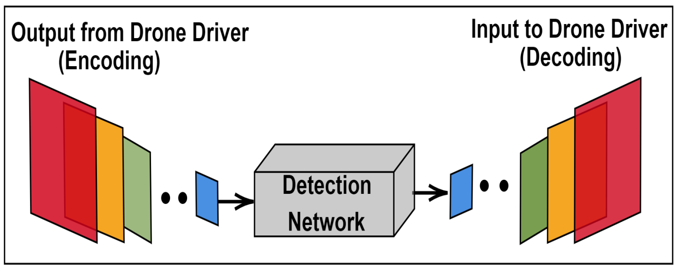


Fig. 4.2 Encoding-Decoding of images

The encoding-decoding process is done to re-arrange all the pixels from the drone camera to fit into the input layer of the fused YOLO. The fused-YOLO receives images from the image acquisition node as input, the targets/weeds are detected and a bounding box is assigned for each weed detected as seen in Fig. 4.3.

Centre Pixel Extraction

After each detection, the centre pixel of the detection bounding box is extracted. It is assumed that the centre of the bounding box coincides with the centre of the weed whose position is to be estimated as in Fig. 4.3. This location in the image frame is converted to world

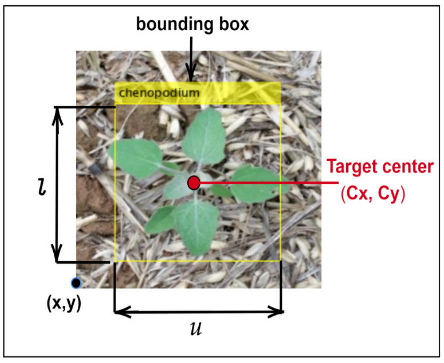


Fig. 4.3 Bounding box extraction. l and u are the length and width of the bounding box respectively. (x, y) represents the origin of the bounding box. (C_x, C_y) represents the target centre.

coordinates as the geometric centre of the weed.

$$\begin{aligned} C_x &= x_p + \frac{u}{2} \\ C_y &= y_p + \frac{l}{2} \end{aligned} \tag{4.4}$$

where C_x and C_y are the centre pixel coordinates, x_p and y_p are the origin points of the bounding box, l and w are the length and width of the bounding box. All the variables are updated with each detection made.

Calculation of Bearing Angles

Provided the frontal camera orientation and pointing axis is known, using the Parrot drone on-board IMU and the odometry information, the centre pixels are converted to bearing angles (α_1, α_2, β_1, β_2,....) from the camera pointing axis to a vector that passes through the targets and the focal point as shown in Fig. 4.4. Afterwards, these angles are converted into the overall azimuth and elevation angles (σ_1 and θ_1, σ_2 and θ_2....) for each target as depicted in Fig. 4.5 through a sequence of conversions (camera frame to drone's body frame and to the world frame) where $r_1, r_2 ... r_n$ are depths from drone to targets. From Fig. 4.4 and 4.5, we can deduce the following for targets 1 and 2:

$$\tan \sigma_1 = \frac{rx_1}{ry_1}, \ \tan \sigma_2 = \frac{rx_2}{ry_2} \tag{4.5}$$

Also,

$$\tan \theta_1 = \frac{rz_1}{\sqrt{(rx_1)^2 + (ry_1)^2}}, \ \tan \theta_2 = \frac{rz_2}{\sqrt{(rx_2)^2 + (ry_2)^2}} \tag{4.6}$$

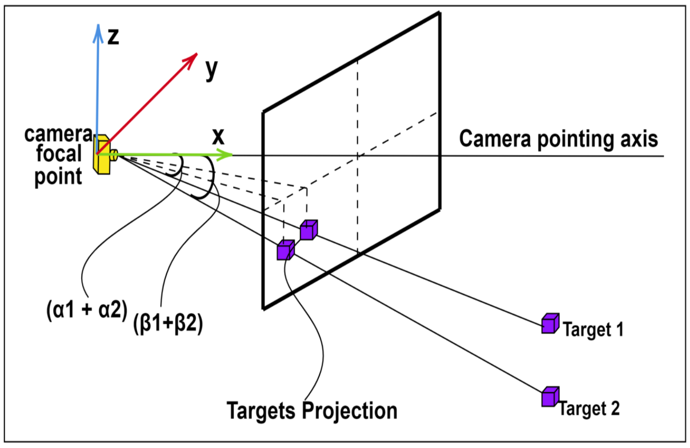


Fig. 4.4 Image projection figure where α_1, α_2, β_1, β_2 represent the bearing angles from the camera pointing axis to a vector that passes through the targets and the focal point.

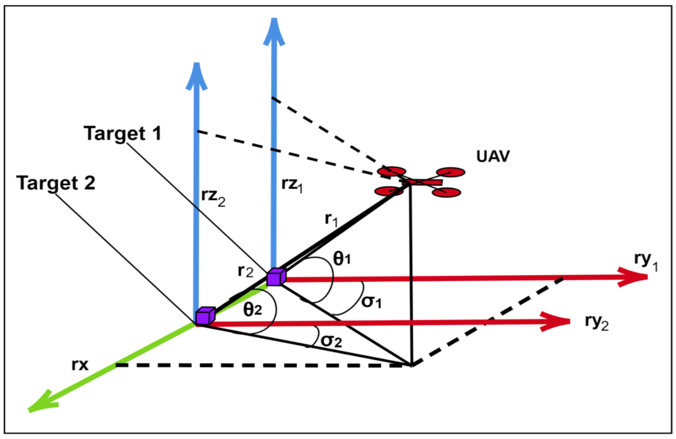


Fig. 4.5 Obtained azimuth and elevation angles σ and θ for targets. $r_1, r_2 ... r_n$ are depths from drone to targets expressed in rx, ry and rz directions.

Unscented Kalman Filtering

The Unscented Kalman filter is a better estimator than the Extended Kalman Filter where the state distribution is approximated by a Gaussian random variable and propagated analytically through first-order linearization of the non-linear system. This can introduce errors and lead to sub-optimal performance (Wan and Van Der Merwe, 2000). The UKF model constitutes of firstly the time update step and then the measurement update step. The time update encompasses the weight and the sigma points calculations. The measurement update utilises the sigma points to generate covariance matrices and Kalman gain respectively (Ruiz and Aouf, 2017).

As the drone follows a prescribed trajectory, different bearing measurements for the position of the target are acquired and these measurements are fused in a UKF framework. The nonlinearity in the azimuth and elevation measurements (σ and θ) limits the performance of the standard linear Kalman filters even for stationary targets such as weeds. The following is the system's dynamics:

$$\begin{aligned} X_{k+1} &= \Phi_{k+1,k} X_k + \lambda_k \\ Z_k &= h(X_k) + M_k \end{aligned} \tag{4.7}$$

Here, $X_k, X_{k+1} \in \mathbf{R}^3$ are the true target positions in ground fixed frame $X = [X^{(g)}\ Y^{(g)}\ Z^{(g)}]^T$ at time instants k and $k+1$, respectively. The output $Z_k = [\sigma,\ \theta]^T \in [0, 2\pi] \times [0,\ \frac{\pi}{2}]$ is the bearing angle at time k. $h(X_k)$ is defined in (4.10). $\Phi_{k+1,k}$ is the state transition matrix of the system from the time k to $k+1$. λ_k and M_k are the process and measurement noise, respectively, which are uncorrelated to Gaussian white noises with zero means and covariances μ_k and ψ_K, respectively, i.e., $(\lambda \sim \mathcal{N}(0, \mu_k)$ and $M_k \sim \mathcal{N}(0, \psi_k))$. The process model is a 3x3 identity matrix since the targets are stationary, therefore the process noise is a zero matrix:

$$\phi_{k,k-1} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \mu_k = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \tag{4.8}$$

The measurement covariance matrix is:

$$\psi_k = \begin{bmatrix} \sigma_1^2 & 0 \\ 0 & \sigma_1^2 \end{bmatrix} \tag{4.9}$$

From Fig. 4.6 we can deduce that: $r_{nx} = a_x - b_{nx}$, $r_{ny} = a_y - b_{ny}$ and $r_{nz} = a_z - b_{nz}$. Also, $a_k = [a_x\ a_y\ a_z]_k^T$ is the position of the UAV, $b_{nk} = [b_{nx}\ b_{ny}\ b_{nz}]_k^T$ are the targets positions and $r_{nk} = [r_{nx}\ r_{ny}\ r_{nz}]_k^T$ are the relative vectors between the UAV and target.

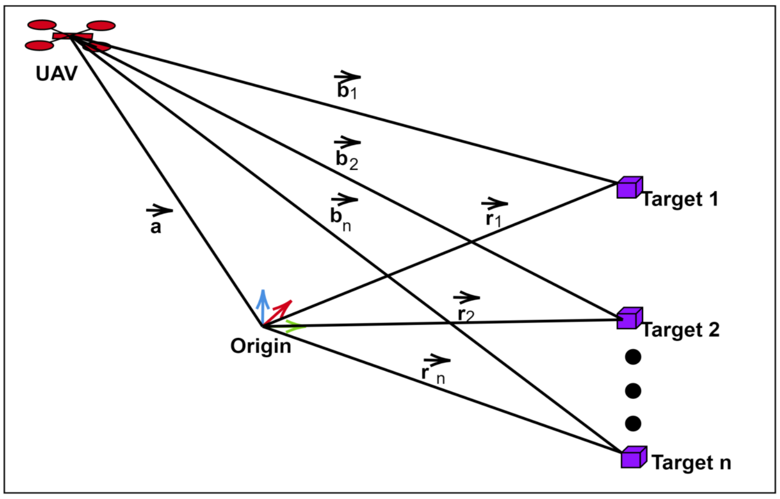


Fig. 4.6 Vector representations of UAV's and targets positions

The measurement model is based on the azimuth angle σ and the elevation θ which are given for target n as follows:

$$Z_{nk} = \begin{bmatrix} \sigma_n \\ \theta_n \end{bmatrix} = \begin{bmatrix} \tan^{-1}(\frac{r_{nx}}{r_{ny}}) \\ \tan^{-1}(\frac{r_{nz}}{\sqrt{(r_{nx})^2+(r_{ny})^2}}) \end{bmatrix} \tag{4.10}$$

The time update: This process includes the calculation of the sigma points and their weights and finally obtaining the time update equations after Cholesky decomposition. We define the weights as in (Ruiz and Aouf, 2017):

$$\begin{aligned} W_1 &= \frac{\zeta}{n+\zeta} \\ W_i &= \frac{1}{2(n+\zeta)} \end{aligned} \tag{4.11}$$

where $i = 1,2,...n$ and n is the state vector dimension which is 3, and ζ is an arbitrary constant assigned to be 0. The sigma points at time k can be calculated as:

$$\begin{aligned} S_{k-1} &= chol((n+\zeta)P_{k-1}) \\ X_{(0)} &= \hat{x}_{k-1} \\ X_{(i)} &= \hat{x}_{k-1} + S_{k-1}^{(i)} \end{aligned} \tag{4.12}$$

Similarly,

$$\begin{aligned} X_{(i+n)} &= \hat{x}_{k-1} - S_{k-1}^{(i)} \\ X_{k-1} &= [X_{(0)} X_{(1)} ... X_{(2n)}] \end{aligned} \tag{4.13}$$

where $i = 1,2,...n$. $S^{(i)}$ is the ith row vector of S and $chol$ means Cholesky decomposition. Finally, the time update equations will be:

$$\begin{aligned} \hat{X}_{\bar{k}} &= \sum_{i=0}^{2n} W_i f(X_i) \\ P_{\bar{k}} &= \sum_{i=0}^{2n} W_i \{f(X_i) - \hat{X}_{\bar{k}}\}\{f(X_i) - \hat{X}_{\bar{k}}\}^T + \mu_k \end{aligned} \tag{4.14}$$

Measurement update: The augmented sigma points can be obtained as:

$$\begin{aligned} S_{\bar{k}} &= chol((n+\zeta)P_{k-1}) \\ X_{(\bar{0})} &= \hat{x}_{\bar{k}} \\ X_{(\bar{i})} &= \hat{x}_{\bar{k}} + S_k^{(i)} \end{aligned} \tag{4.15}$$

Similarly,

$$\begin{aligned} X_{(i\bar{+}n)} &= \hat{x}_{\bar{k}} - S_{\bar{k}}^{(i)} \\ X_{\bar{k}} &= [X_{(\bar{0})} X_{(\bar{1})} ... X_{(\bar{2n})}] \end{aligned} \tag{4.16}$$

where $i = 1, 2, ...n$.

$$\hat{z}_{\bar{k}} = \sum_{i=0}^{2n} W_i h(X_i) \tag{4.17}$$

Finally the measurement covariance and the Kalman gain are calculated as:

$$\begin{aligned} &P_{\bar{z}} = \sum_{i=0}^{2n} W_i \{h(X_i) - \hat{z}_{\bar{k}}\}\{h(X_i) - \hat{z}_{\bar{k}}\}^T + \psi_k \\ &G_k = P_{xz} P_z^{-1} \end{aligned} \tag{4.18}$$

The final estimated state and it's covariance are:

$$\begin{aligned} \hat{X}_k &= \hat{X}_{\bar{k}} + G_k(z_k - \hat{z}_k) \\ P_k &= P_{\bar{k}} - G_k P_z G_k^T \end{aligned} \tag{4.19}$$

The position of the targets in the ground frame is the output from the estimator.

4.4 Gazebo Simulation and Experimental Set-up

4.4.1 Gazebo Simulation

Gazebo is an open-source software for the simulation of robotics platforms. It allows users to make and construct their simulation environment and depicts approximately how the real-world scenario/result will look like. The dynamics of the robot parts are assigned to a model depending on the behaviour intended for the part. It also supports the ROS plug-in. A user can easily link written code mostly in C++ or Python to the gazebo environment. For the simulation-level verification experiments, a Gazebo environment is used. A model of the Parrot AR drone is developed in Gazebo (Engel et al., 2014), as demonstrated in Fig. 4.7. The drone is equipped with all the sensors (monocular camera, rotors, ultrasonic sensors, etc) as in the real platform. The properties of the sensors are assigned to match real characteristics as best as possible.

For this simulation, ROS is used together with Gazebo. In addition to the drone and sensors model, the ROS nodes responsible for the detection and classification of the weed using Deep Neural Network, centre pixel extraction on the images, calculating the bearing angles, fusing the bearing angles to estimate world coordinates of the weed with UKF, and the trajectory planning and drone driving behave as the same in the real-time. A typical scene in Gazebo is presented in Fig. 4.7. The black dot-like object represents the target/weed.

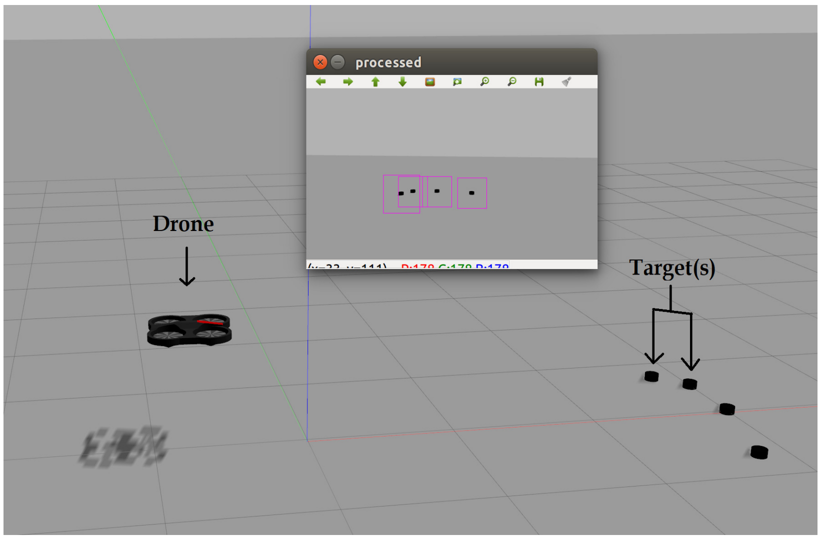


Fig. 4.7 Gazebo environment set-up showing the drone and the targets. The black dot-like objects labeled as targets represent the weeds

4.5 Experimental Set-up

The implementation experiment we conducted has a set-up of a drone shown in Fig 4.8, a workstation, an optitrack tracking system 4.9, and the target to estimate(weed). Firstly, the algorithm for the weed detection in Chapter 3 was converted to C++ from MATLAB with the help of MATLAB coder similar to Section 3.10.1.

The weights of the neural network are exported as a static library, A function format is used to easily access the weights from a script. The target hardware to run the neural network

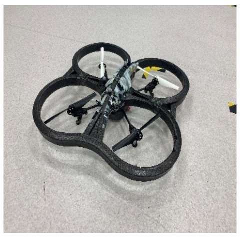


Fig. 4.8 Parrot Drone

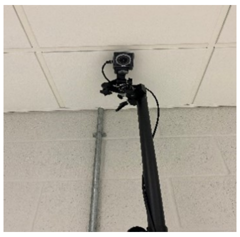


Fig. 4.9 Optitrack Tracking Camera

is a Central Processing Unit (CPU). Normally, the CPU will not support the neural network without a supporting library called MKLDNN (Zarukin, 2014). The MKLDNN library is linked so that the neural network can run on the CPU. Alternatively, the CUDA and CUDnn libraries can be used with a GPU to run the network. Even after deploying the pipeline in C++, it can not run straightaway. This is because the way MATLAB perceives an image input differs from the deployment method in C++. To receive a continuous stream from the drone camera, An encoding and decoding operation (see Fig. 4.2) must be executed to make the C++ capable of understanding and interpreting the input image. Another problem to cater for is integrating the Optitrack systems. The Parrot drone is designed as a wireless network host likewise the Optitrack system. It is not possible to connect the workstation to the drone and at the same instance connect to the optitracks to get position feedback. A solution used is to alter the internal configuration of the drone to serve as a client to a network. This way, a router can be used as a Local Network onto which the Optitrack, drone and workstation can connect and interchange data as in Fig 4.10. Target devices can easily be queried by switching between IP addresses.

An actual parrot drone shown in Fig 4.8 is subjected to a trajectory while the onboard monocular camera is utilised to detect these weeds. The trajectory parameters are selected such that the targets are covered in the FOV of the drone. A typical trajectory for four targets

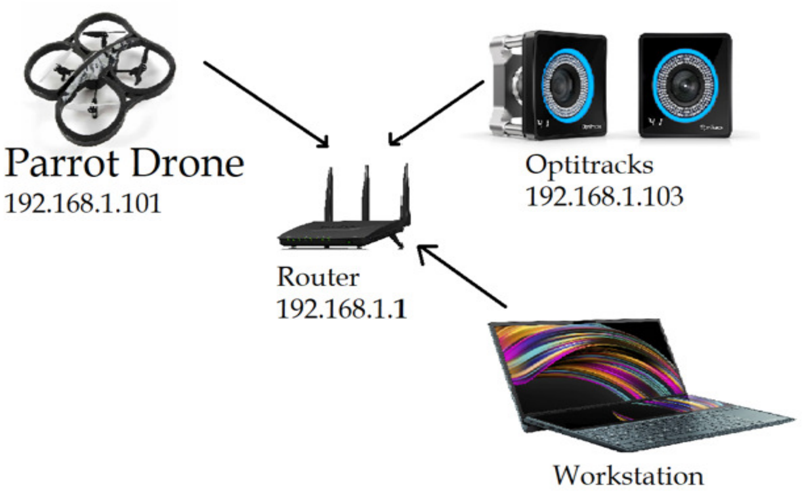


Fig. 4.10 Network Connection Set-up

is seen in Fig. 4.11 using a major axis radius a = 0.6m and minor axis radius b = 0.4m with a height of 1m. The real-time experiment differs from the simulation. As for the simulation,

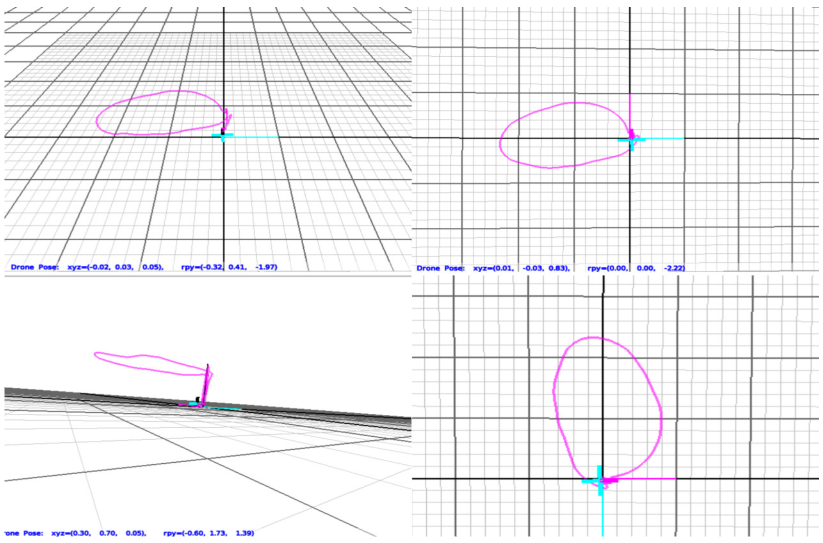

Drone Pose: xyz=(-0.02, 0.03, 0.05), rpy=(-0.32, 0.41, -1.97)
Drone Pose: xyz=(0.01, -0.03, 0.83), rpy=(0.00, 0.00, -2.22)
one Pose: xyz=(0.30, 0.70, 0.05), rpy=(-0.60, 1.73, 1.39)

Fig. 4.11 Obtained trajectory using a major axis radius a = 0.6m and minor axis radius b = 0.4m.

mapping between the drone driver and the gazebo model is made to retrieve the ground truth position of the drone. However, in the real-world experiment, mapping between the drone and the real drone platform itself is made. The ground truth information received from the gazebo formally is replaced with the tracking system to obtain the ground truth position of the weeds.

Connecting to the tracking system is not enough to establish a pipeline that receives position updates, a supporting package is used to receive the broadcasted positions from the tracking system so it can be used as a ROS topic and can be subscribed by any node. The real-time experiment is carried out on i7 Core CPU. It took approximately 45 seconds to complete the estimation which includes weed detection and localization as well. All the CPU cores were utilized with an average utilisation factor of 90% while running the fused YOLO. The frequency of the CPU was maintained at 2435MHz.

The tracking system while tracking the drone updates the workstation with the real-time position of the drone. As per the measurement model in Eq. 4.10, It is required to vary r to continuously obtain azimuth and elevation angles. An elliptic trajectory is maintained to obtain a varying r to use for the estimation.

4.6 Results

For both simulation and experimental works, the initialisation of the filter was done arbitrarily, however, the initial state estimate and its covariance are taken as follows:

$$\hat{X}_0 = \begin{bmatrix} 20 \\ 20 \\ 20 \end{bmatrix}, P_0 = \begin{bmatrix} 45 & 0 & 0 \\ 0 & 45 & 0 \\ 0 & 0 & 45 \end{bmatrix}, \zeta = 0 \tag{4.20}$$

Targets/weeds are placed at different ground truth positions. The drone is placed at $[x, y] = [0.00, 0.00]m$ for simulations and experiments. For the simulations, the ground truth is obtained from the Gazebo simulation environment. While in the experiments, both drone and targets are placed within the volume of the tracking system so that feedback on the ground truth positions can be received. The drone makes a trajectory within the volume while estimating the relative position of the placed targets. The ground truth of targets during the experiment is different from the simulation to accommodate the tracking range of the

Optitrack system. This will not have any effect on the estimation but proves the robustness of the estimation at different ground truth positions.

4.6.1 Simulation Results

The simulation results were analysed by comparing the estimated positions with the ground truth position. Weeds were placed at ground truth positions $[x, y] = [4.00, 1.00]$, $[3.50, 0.50]$, $[3.00, 0.00]$, $[2.50, -0.50]m$. For the longitudinal tests, the estimates of x component of the ground truth i.e $[x] = [4.00], [3.50], [3.00], [2.50]m$ are taken. In Fig. 4.12, the dashed lines represent the ground truth position while the continuous lines represent the estimated position. The results show the convergence of the estimator along the x-axis of the ground frame: the estimated positions are obtained after 35 seconds of the estimation process which is decent compared to the literature (Ponda et al., 2009; Ruiz and Aouf, 2017). Fig. 4.12 also shows the changes in the estimation error with time. The error is measured as the absolute difference between the ground truth position and the estimated position. The error approaches zero as the estimator receives updates.

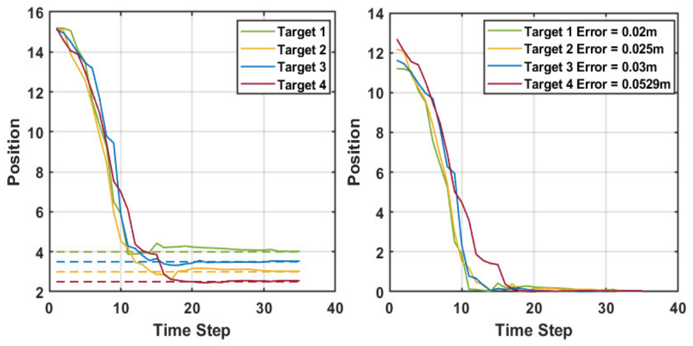


Fig. 4.12 Coordinate estimation and error (simulation tests along the x-axis). The dashed lines represent the ground truth while the continuous lines are the estimations.

The 2D localisation of the weed is performed simultaneously. For the lateral tests, we estimate the y component of the ground truth i.e. $[y] = [1.00], [0.50], [0.00], [-0.50]m$. The results are demonstrated in Fig. 4.13. The performance of the proposed solution along the lateral axis is satisfactory and gets better over time, as expected.

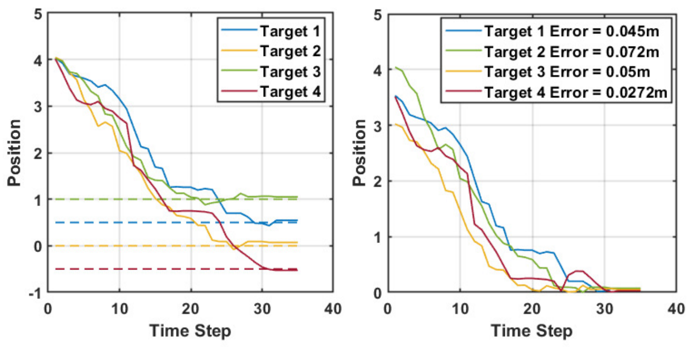


Fig. 4.13 Coordinate estimation and error (simulation tests along the y-axis). The dashed lines represent the ground truth while the continuous lines are the estimations.

To further verify the robustness of the estimator, the same experiment was conducted with different ground truth positions. Table 4.1 presents the experiment scenario and the results. The results prove the success of the position estimation with no sophisticated sensors in the simulation environment. An average error of 0.066m was obtained along the x direction, and 0.082m along the y direction.

Table 4.1 Additional simulation results using YOLOv4 with targets placed at positions [x,y] = [4.00,0.22], [3.00,0.036], [3.00, 0.50], [3.50,-0.36] and [2.80, -0.06]. The error of estimation is the difference between the ground truth and the estimated positions.

Takes	Ground Truth (x)	Estimated (x)	Error(x)
1	4.00	3.85	0.15
2	3.00	3.10	0.10
3	3.00	3.03	0.03
4	3.50	3.47	0.03
5	2.80	2.82	0.02
Takes	**Ground Truth (y)**	**Estimated (y)**	**Error(y)**
1	0.22	0.127	0.093
2	0.036	0.025	0.011
3	0.50	0.610	0.11
4	-0.36	-0.51	0.15
5	-0.06	-0.016	0.044

4.6.2 Experimental Results

The results from the experiments are presented under the following categories:

Detection Score

The accuracy of realtime detection has an effect on the overall estimation performance since the centre of the bounding box of the detected weeds is assumed to match the geometric centres of the targets. The detection score evaluates how well a bounding box is assigned to a target $C_b = C_t$ where C_b and C_t are the centres of the bounding box and target, respectively so that the bounding box accurately covers the area of the target. (see Equ. 3.6). The lowest detection score recorded for all the targets/weeds detected is 79% and the maximum is 95%. Fig. 4.14 shows the detection scores with their frequencies. The frequency in this context is defined as how many times a particular detection score was obtained throughout the estimation.

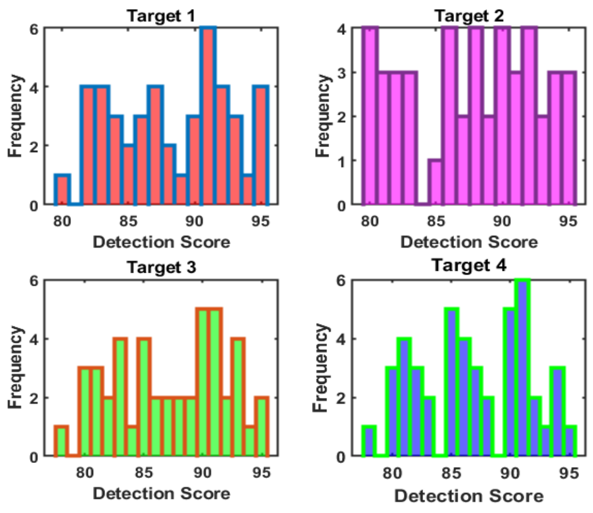


Fig. 4.14 Detection score

Detection Deviation

The detection deviation is defined to indicate how much is the deviation in $C_b \neq C_t$. It provides a clear indication of the error that is introduced to the estimator due to misleading extraction of the centre position of the bounding box. It is defined as the difference between the ideal detection score and the obtained detection score. For the experimental works, the deviation score is limited to 21%, which means that up to 21% deviation in C_b from C_t is considered tolerable for obtaining an accurate estimation. The histogram plot for the deviation is shown in Fig. 4.15.

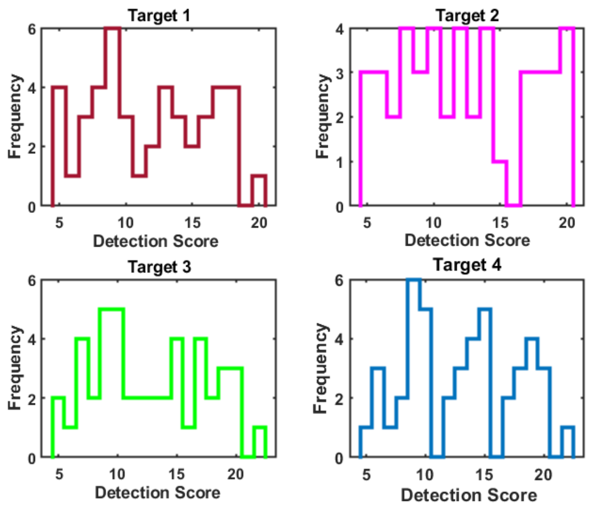


Fig. 4.15 Detection deviation

Position Estimation

As the drone follows its predefined elliptical trajectory, the depth, azimuth and elevation angles vary continuously and the measurements are fused to estimate the positions of targets. Weeds are placed at ground truth positions $[x,y]$ = $[5.00,1.00]$, $[4.50,0.50]$, $[4.00,0.00]$, $[3.80,-0.50]m$. For the longitudinal tests, the estimates of the x component of the position are taken as $[x]$ = $[5.00]$, $[4.50]$, $[4.00]$, $[3.80]m$. The estimated positions and the position estimation errors are shown in Fig. 4.16. For the lateral tests, the y component of the position is estimated and the results are demonstrated in Fig. 4.17.

To show the robustness of the proposed scheme, a test is conducted where the Optitrack can not sufficiently provide feedback to the drone for control. The fixed altitude assumption is violated in these experiments, consequently, the estimator recorded a greater error in these scenarios, as can be seen in Fig. 4.18. Targets were placed at a y coordinate $[y] =$

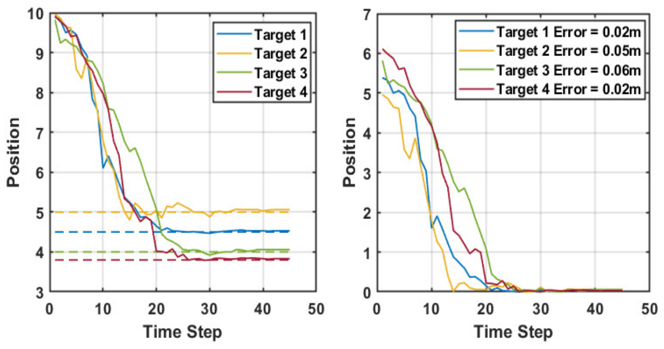


Fig. 4.16 Coordinate estimation and error (experimental tests along the x-axis). The dashed lines represent the ground truth while the continuous lines are the estimations.

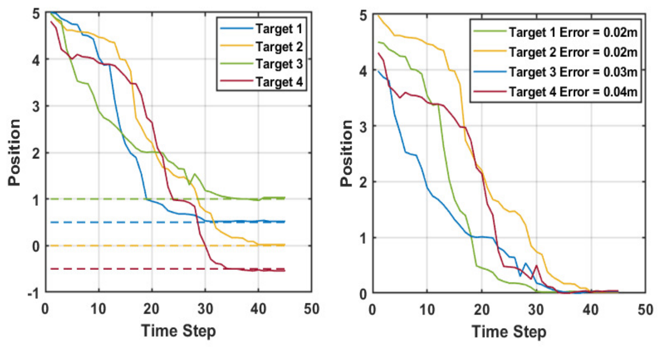


Fig. 4.17 Coordinate estimation and error (experimental tests along the y-axis). The dashed lines represent the ground truth while the continuous lines are the estimations.

$[-0.60], [-1.00]m$ and the drone was placed at initial position $[x, y] = [-1.00, 0.00]m$ such that not all updates will be received because of a limited tracking volume. In other words, the drone can not be tracked at some points along the trajectory. Despite this disturbance, a maximum error of 0.35m was experienced.

More experimental results were obtained by repeating the same experiment with different ground truth positions to verify the robustness. Table 4.2 shows the obtained estimation with the associated error. An average error of 0.056m was obtained along the x direction and 0.113m along the y direction.

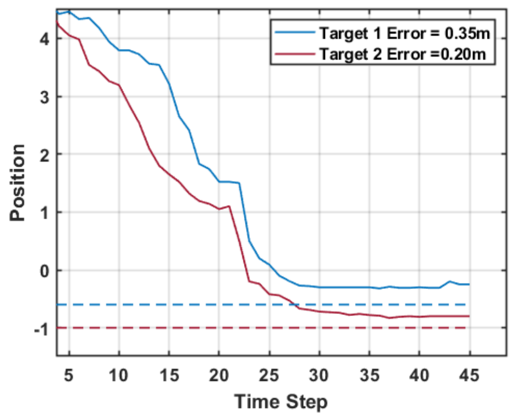


Fig. 4.18 Coordinate estimation and error (experimental tests along y-axis, insufficient optitrack data case). The dashed lines represent the ground truth while the continuous lines are the estimations.

Table 4.2 Additional experimental results using YOLOv4 with targets placed at positions [x,y] = [4.62, 0.00], [4.43, 0.50], [5.12, -0.50], [3.89, 1.00] and [4.04, -1.00]. The error of estimation is the difference between the ground truth and the estimated positions.

Takes	Ground Truth (x)	Estimated (x)	Error(x)
1	4.62	4.57	0.03
2	4.43	4.45	0.03
3	5.12	5.19	0.07
4	3.89	3.77	0.10
5	4.04	4.09	0.05
Takes	**Ground Truth (y)**	**Estimated (y)**	**Error(y)**
1	0.00	0.05	0.05
2	0.50	0.49	0.01
3	-0.50	-0.26	0.24
4	1.00	1.065	0.065
5	-1.00	-1.20	0.20

Depth Estimation

The depth, r, estimation from the UKF is compared with the depth measured using the positions of the UAV and the target obtained with the Optitrack system. Since the UAV's height H is known, the depth will be equal to $\sqrt{(D)^2+(H)^2}$, where D is the difference of the target's position and the UAV's position along the direction which of the camera frame. Flying at a fixed height of $1m$ will reduce the depth to $\sqrt{(D)^2+1}$. Fig. 4.19 shows the convergence of the estimated depth and the calculated depth. The calculated depth is not constant as the drone is subjected to an elliptical trajectory thus, a varying elevation angle θ will be received along the trajectory. It is observed that with time, the depth estimation gets better and resembles the acceptable level.

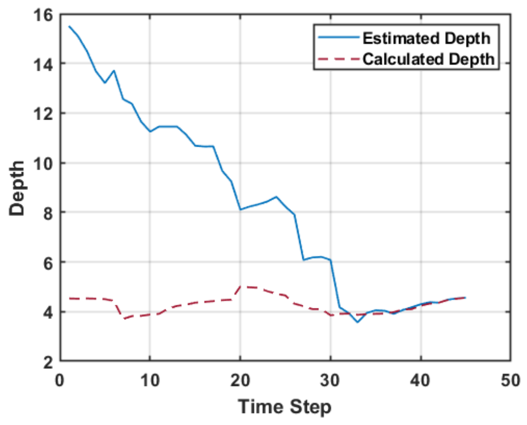


Fig. 4.19 Depth estimation

The experimental results converge after 45 seconds while the simulation results converge after 35 seconds. This is majorly due to the latency as the image updates are transported over a network in the experimental setup. On the other hand, the simulation setup assumes an ideal world with no update delay.

Detection Score Comparison

Our proposed pipeline for the relative pose estimation can be utilised with different YOLO versions by simply substituting the detection end of the pipeline. This adds to the value of the pipeline as it shows how flexible the pipeline can be adapted to other versions of YOLO. This can be done by truncating the final layers of ResNet-50 and utilising the final activation layer of the RestNet as the feature extraction of the preferred YOLO version as earlier explained in Chapter 3 (see Fig 3.7). Provided a detection is made and a bounding box is assigned to the target, the detected target's position can be estimated. However, there can be a slight deviation in detection scores across different YOLO versions which prompted this analysis. Recall that the detection score may have an effect on the estimation.

Newer versions of YOLO such as YOLOv4 may have better accuracy and Frames per Second (FPS). However, these properties may not significantly increase the overall accuracy of the estimation since the detection is performed at regular time steps. Nonetheless, the most sensitive parameter is the detection score which can introduce errors to the estimation. A better detection score will result in better estimation. We define the detection score as how well the centre of the bounding box aligns with the centre pixels of the target. The detection scores obtained using both YOLOv2 (Redmon et al., 2016b) and YOLOv4 are compared across 45-time steps for each target as seen in Fig. 4.20 - 4.23. The overall Average Detection Score (ADS) for the four targets using YOLOv4 is 87.505% while with YOLOv2 it is 86.5%. Although both ADS fall within an acceptable range for this experiment, YOLOv4 is expected to provide a slightly more accurate result than YOLOv2 since it has a better detection score.

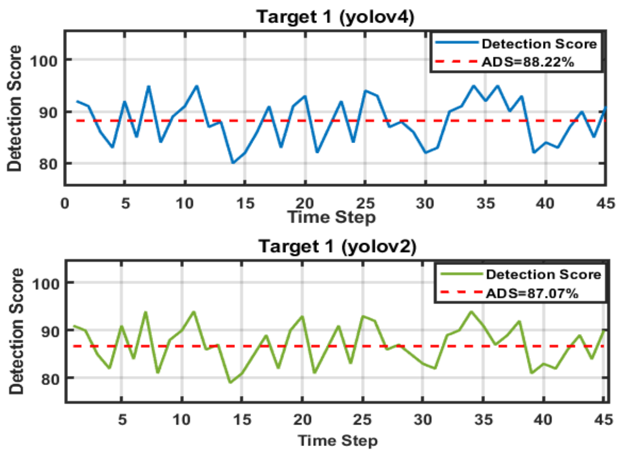


Fig. 4.20 Detection score comparison between YOLOv2 and YOLOv4 for target 1

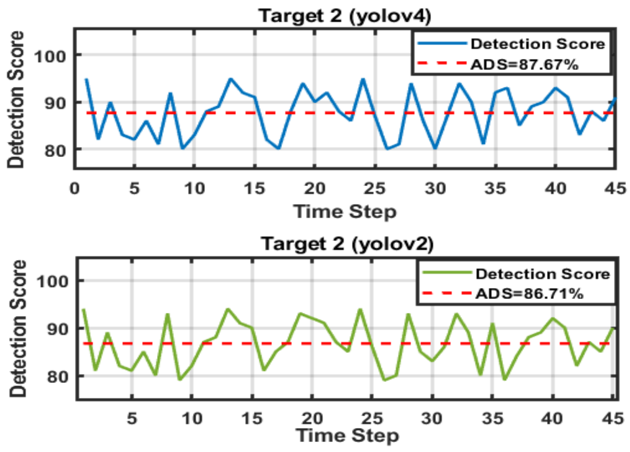


Fig. 4.21 Detection score comparison between YOLOv2 and YOLOv4 for target 2

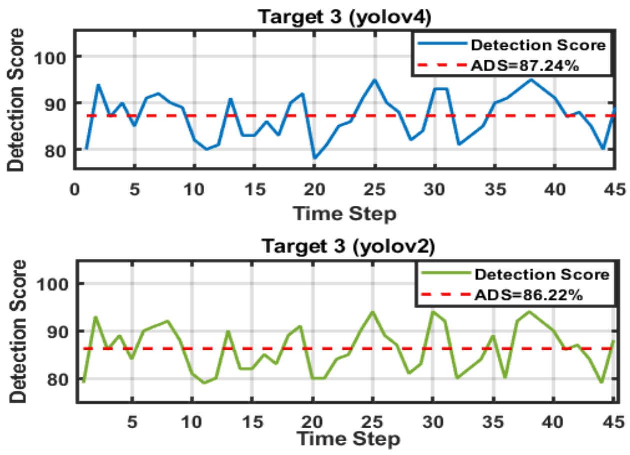


Fig. 4.22 Detection score comparison between YOLOv2 and YOLOv4 for target 3

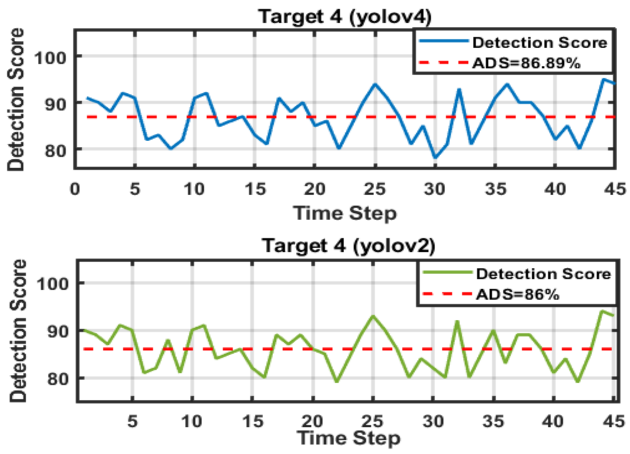


Fig. 4.23 Detection score comparison between YOLOv2 and YOLOv4 for target 4

Furthermore, the effect of the ADS on the estimation can be observed in Table 4.3. Targets are placed at known ground truth positions and are tested us both versions of YOLO. It can be observed that the YOLOv4 has a better estimation due to having a better ADS than the YOLOv2. However, the difference is not much and we can conclude that the version YOLO used does not significantly affect the estimation but can slightly improve the estimation based on a better ADS.

Table 4.3 Additional experimental results comparing YOLOv2 (v2) and YOLOv4 (v4) estimations with targets placed at Ground Truth positions (GT) [x,y] = [4.62, 0.00], [4.43, 0.50], [5.12, -0.50], [3.89, 1.00] and [4.04, -1.00].

Takes	GT(x)	Estimated v2 (x)	Error v2 (x)	Estimated v4 (x)	Error v4(x)
1	4.62	4.58	0.04	4.57	0.03
2	4.43	4.45	0.03	4.45	0.03
3	5.12	5.21	0.09	5.19	0.07
4	3.89	3.78	0.11	3.77	0.10
5	4.04	4.09	0.05	4.09	0.05
Takes	**GT (y)**	**Estimated v2 (y)**	**Error v2 (y)**	**Estimated v4 (y)**	**Error v4(y)**
1	0.00	0.06	0.06	0.05	0.05
2	0.50	0.49	0.01	0.49	0.01
3	-0.50	-0.23	0.27	-0.26	0.24
4	1.00	1.07	0.07	1.065	0.065
5	-1.00	-1.20	0.20	-1.20	0.20

Network Performance During Position Estimation

The performance of the proposed network during estimation is evaluated for both indoor and outdoor scenarios. Three classes of weeds namely: *Crassulaceae, Astroloba* and *Piperaceae* are used in the experiments. The obtained final training loss was 0.032 for the outdoor training as seen in Fig 4.24. The outdoor validation in Fig. 4.25 shows the Average Precision (AP) obtained for these weed classes. *Crassulaceae* obtained an AP of 0.8643, *Astroloba* obtained an AP of 0.9362, and *Piperaceae* obtained an AP of 0.9432 across 443 frames. Samples of the detection using the Fused-YOLO network are displayed in Fig. 4.26. The bounding boxes as seen in most cases are corresponding to the target's centre which will facilitate better position estimation.

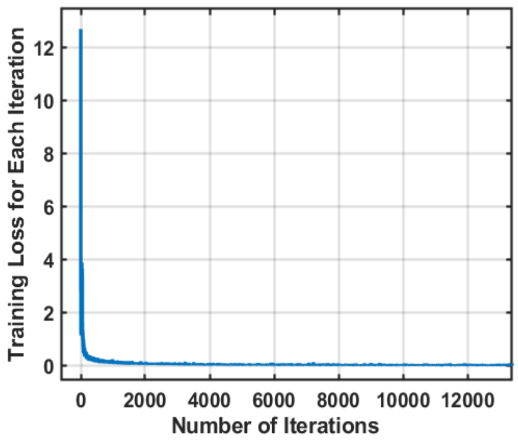


Fig. 4.24 Training Loss per iteration for Fused-YOLO Network in an outdoor setting

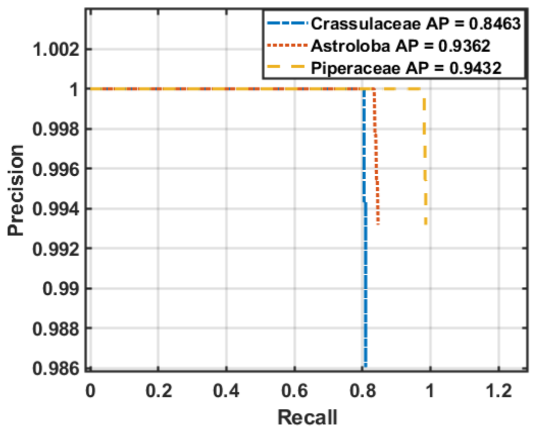


Fig. 4.25 Precision/Recall results for *Crassulaceae, Astroloba*and *Piperaceae* with their respective AP in an outdoor setting

For the indoor setting, a final training loss of 0.0203 was obtained as shown in Fig. 4.27. The validation results from Fig. 4.28 show the AP of *Crassulaceae* class at 0.8658, *Astroloba* at 0.8973, while *Piperaceae* obtained an AP of 0.9133 evaluated across 315 frames.

Another experiment was conducted to estimate the positions of the different classes of weeds concurrently. The different classes of weeds (*Crassulaceae, Astroloba* and *Piperaceae*) were placed at ground truth positions at $[x,y] = [2.7,0.2]$, $[2.8,0.8]$ and $[2.48,0.5]$ respectively.

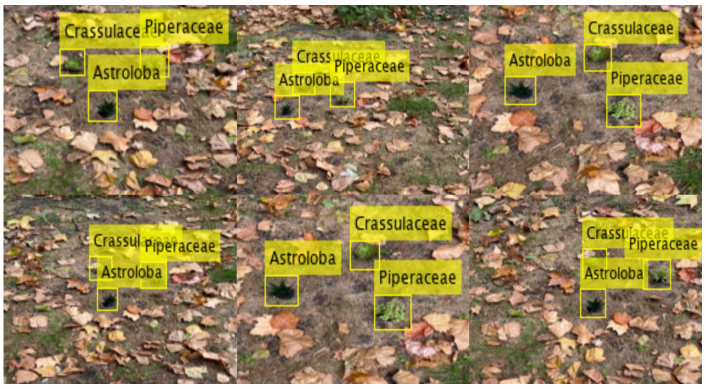


Fig. 4.26 Qualitative Result samples from Fused-YOLO Network in an outdoor setting

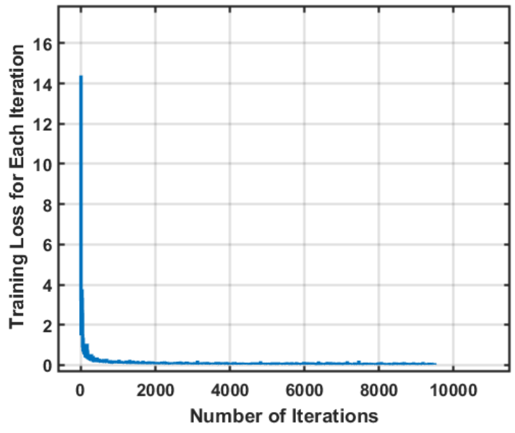


Fig. 4.27 Training Loss per iteration for Fused-YOLO Network in an indoor setting

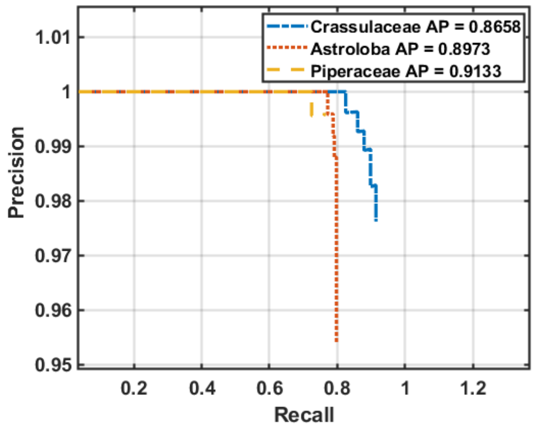


Fig. 4.28 Precision/Recall results for *Crassulaceae, Astroloba*and *Piperaceae* with their respective AP in an indoor setting

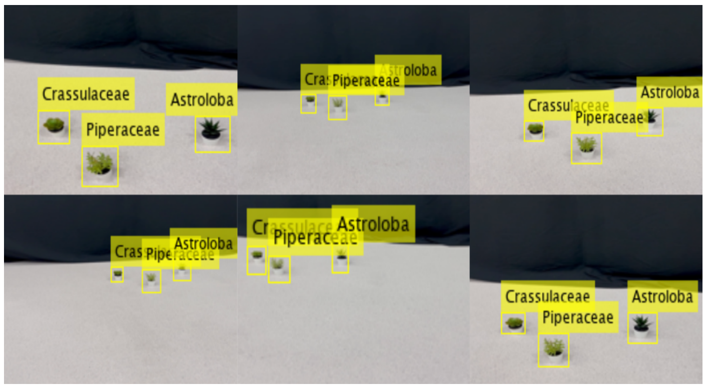


Fig. 4.29 Qualitative Result samples from Fused-YOLO Network in an indoor setting

The error obtained for each class is shown in Fig. 4.30. An error of $[x, y] = [0.05m, 0.055m]$ was observed for *Crassulaceae* class while for *Piperaceae* and *Astroloba* classes, the errors are found to be $[x, y] = [0.025m, 0.04m]$ and $[x, y] = [0.042m, 0.043m]$ respectively. These results further validate that the proposed pipeline can effectively estimate the positions of different classes (types) of weeds.

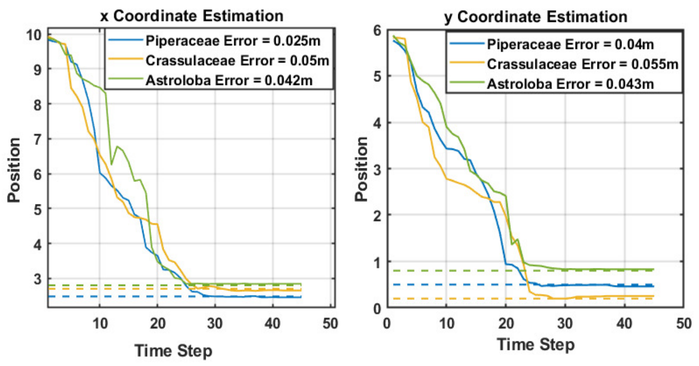


Fig. 4.30 Coordinate estimation [x,y] (experimental tests with different classes of weeds). The dashed lines represent the ground truth while the continuous lines are the estimations.

4.7 Conclusion and Future Work

This chapter discusses the implementation of relative position estimation for multiple targets (weeds) by the combination of UKF with a deep neural network. It addresses the use of sophisticated algorithms for position estimation and detection of weeds while presenting a faster and more reliable result with good accuracy using affordable sensors. It extends to not only using bounding boxes for detection but utilizing them for position estimation.

In the proposed solution for weed detection, an affordable UAV platform with a monocular camera is used. Weeds are detected and classified using the trained neural network (similar to the proposed one in Chapter 3) and the detection boxes are utilized to extract the centre of the target using the image data from the UAV platform which performs an elliptic trajectory and thus, forms the basis for the varying bearing angles for UKF estimation. The UKF utilizes noisy azimuth and elevation angles to perform the estimation.

The simulation results converge after 35 seconds while the experimental results converge after 45 seconds. The detection score is 87.5% on average. The overall average estimator error is (x= 0.056m, y= 0.0703m). The proposed method is able to achieve multiple targets (weed) position estimation with a lesser error margin using an off-the-shelf platform without requiring any sophisticated or additional devices or sensors. The estimation error is measured from the weed's centre and most detectable weeds have a cross-section of up to or more than 5-8cm. Thus, this gives a good margin for the localisation of the weeds. Also, these positions are initially estimated positions, mechanical weeding arms or sprayers are usually accompanied by a camera to perform visual servoing (post-processing) to fine-tune the exact positions of the target and hence these results are satisfactory for this mission.

For future works, trajectory optimisation can be investigated for a better field of view so more targets/weeds can be estimated at once. Cooperative estimation can be investigated using multiple homogeneous and heterogeneous platforms for faster convergence of the position estimator.

Chapter 5

TransPose: A Transformer-based 6DoF Object Pose Estimation Network with Depth Refinement

Chapter abstract

The position estimation scheme proposed in Chapter 4 is limited to aerial platforms and may be challenging especially if the problem requires more than 3-DoF pose estimation. In this chapter, TransPose, an improved Transformer-based 6D pose estimation with a depth refinement module is introduced. The architecture takes an RGB image as input with no additional modalities (depth or thermal). The architecture encompasses an innovative lighter depth estimation network architecture that estimates depth from an RGB image using a feature pyramid. A transformer-based detection network with additional prediction heads is proposed to directly regress the object's centre and predict the 6 DoF pose of the target. A novel depth refinement module is then used alongside the predicted centres, poses and depth patches to refine the accuracy of the estimated 6-DoF pose. Results are extensively compared with other state-of-the-art methods and analysed for fruit-picking applications. As part of this work, the first-ever novel fruit dataset with multiple modalities and tailored specifically for pose estimation is proposed.

5.1 Motivation

Recall that this thesis discussed 2D relative position estimation $[x,y]$ in Chapter 4. This is achieved by subjecting a UAV to a trajectory. However, in some agriculture problems, performing a trajectory for the relative position is not possible. These applications do not have the luxury of space or manoeuvrability to easily conduct a trajectory and furthermore estimate the position of the target. Also, the majority of agricultural activities will require more than just the 2D position information. Thus, there is an urgent need for a system that can estimate 6DoF relative pose (translation and rotation) without conducting a trajectory.

As demand for robotics manipulation application increases, accurate vision-based 6DoF pose estimation becomes essential for autonomous operations. Convolutional Neural Networks (CNNs) based approaches for pose estimation have been previously introduced. However, the quest for better performance still persists especially for accurate robotics manipulation as is the case in the Agri-robotics domain.

Generally, 6DoF object pose estimation is a crucial topic to address especially in the robotics domain. The ability to perceive the position of an object from a single RGB image finds application in areas such as robotics for grasping tasks (Zhu et al., 2014), autonomous driving (Menze and Geiger, 2015) and robotics for virtual and augmented reality applications (Marchand et al., 2015). This problem, however, comes with several challenges such as object appearance and texture, lighting conditions and object occlusion (Xiang et al., 2017).

Conventionally, the 6-DoF pose estimation problem is formulated as a feature mapping problem where feature points of 3D objects are matched on 2D images (Collet et al., 2011; Lowe, 1999; Rothganger et al., 2003). However, these methods are unable to detect features on smooth objects with minimum or no texture. The introduction of additional modalities such as depth data have been used to solve the problem of features on texture-less objects (Bo et al., 2014; Brachmann et al., 2014; Hinterstoisser et al., 2012). However, this requires more inputs in the form of RGB-D images. With the emergence of CNN, some research leveraged this powerful tool as part of their pipeline to estimate 6-DoF poses (Wang et al., 2019b; Xiang et al., 2017). Transformer-based models are emerging and proving to be more efficient than CNNs (Carion et al., 2020; Dosovitskiy et al., 2020; Khan et al., 2022; Touvron et al., 2021). Thus, few pipelines adopting transformer-based models for 6-DoF pose estimation in the quest for better accuracy (Amini et al., 2021; Beedu et al., 2022; Jantos et al., 2023) exist.

In this chapter, a novel 6DoF object pose estimation architecture is presented aiming at improving the accuracy in comparison to the existing methods. TransPose: an improved transformer-based 6-DoF pose estimation network with a novel depth refinement module is introduced. The objective is to get better 3D translations and rotation estimates from a single RGB image input. For initial estimations, Detection Transformer (DETR) framework (Carion et al., 2020) is adapted to directly regress the centre of the target object. Furthermore, an image patch of the target object is obtained so that translation and rotation can be directly regressed by additional prediction heads on the DETR (Amini et al., 2021). Indeed, feed-forward heads are added to regress the two components of the 6DoF pose (3D translation and 3D rotation). A novel depth refinement module is also introduced in our estimation pipeline to increase the accuracy of the pose estimation.

The unavailability of a robust dataset for fruit pose estimation introduces a challenge for AI-based fruit pose estimation and hinders development in this direction. This chapter by extension proposes the first-ever and novel fruit dataset with multiple modalities to bridge the dataset gap and perhaps facilitate development in the domain.

5.2 Related Work

Many methods have been proposed to tackle the problem of 6D object pose estimation. Approaches that are non-learning-based rely heavily on object textures for pose estimation. Scale-Invariant Feature Transform (SIFT) features (Lowe, 2004) and Speeded Up Robust Features (SURF) (Bay et al., 2006) are common examples of the classical features used. The SIFT algorithm as used by (Zhang et al., 2014) for pose estimation requires rich texture information. This can be an issue if the objects are textureless. E. Miyake et al. (Miyake et al., 2020) compensated the textureless nature of objects with the colour information to improve the accuracy of the 6DoF pose estimation. The geometric information has also been used to increase the accuracy of estimation (Zhang et al., 2018).

Pose estimation methods that utilise local descriptors define and compute the global descriptors offline. The local descriptor is then computed and matched online with the global descriptor. Pose estimation using Iterative Closest Point (ICP), Oriented Fast and Rotated Brief (ORB) (Rublee et al., 2011), Binary Robust Independent Elementary Features (BRIEF) (Calonder et al., 2010) have been implemented in the past (Akizuki and Aoki, 2018; Guo et al., 2019; Yu et al., 2018). However, these methods are computationally expensive and do not perform well on reflective objects.

We can further group pose estimation methods into template-based methods and features-based methods (Xiang et al., 2017). The advantage of the template-based methods is that they can detect objects without enough textures. Each location of the input image is scanned and matched with a constructed template of the object. The best match is selected based on a similarity score that compares the matched locations (Cao et al., 2016; Hinterstoisser et al., 2011, 2012). These types of methods cannot properly estimate occluded objects since the similarity score will be low.

The feature-based methods utilize 2D-3D correspondences. Features are mapped from the 2D images to the 3D models thereby estimating the 6D poses (Lowe, 1999; Rothganger et al., 2003; Tulsiani and Malik, 2015). This approach handles occluded objects better. However, this is at the expense of rich features in the form of enough texture. Some works have proposed learning feature descriptors to solve the problem of objects with no texture (Doumanoglou et al., 2016a; Wohlhart and Lepetit, 2015), while others regress directly from the 2D image to obtain the 3D correspondence (Amini et al., 2021; Brachmann et al., 2014, 2016; Krull et al., 2015). Without sufficient refinement, these models can obtain relatively low accuracy when dealing with symmetrical objects.

Convolutional Neural Network (CNN) architecture for pose estimation was introduced by (Kendall et al., 2015) to regress 6D pose using RGB image. Limited by depth modality, the task becomes difficult. In an attempt to address this problem, another method proposed the prediction of depth from the 2D image and thus acquire the 3D position of the object (Xiang et al., 2017). Estimating the rotation component can also be a problem using this method due to non-linearity. (Liu et al., 2016; Su et al., 2015; Sundermeyer et al., 2018) separated the rotation component and treated it as a classification problem. This often requires a post-refinement to obtain an accurate estimation. Methods that detect keypoints to estimate 6D pose have been proposed to robustly and efficiently estimate 6D pose. (Rad and Lepetit, 2017) utilised a segmentation technique to isolate the Region of Interest (ROI) and further regressed the keypoints from the ROI. Similarly, (Tekin et al., 2018) utilised the YOLO (Redmon and Farhadi, 2017b) framework for such. However, these methods in the face of occlusion perform poorly. To address this problem, some methods obtain keypoints through pixel-wise heatmaps (Oberweger et al., 2018; Pavlakos et al., 2017). Considering that heatmaps are fixed-size, these methods suffer when the objects are truncated.

Some other methods have considered using models encompassing classical algorithms such as the PnP algorithm to increase the accuracy of estimation (Hu et al., 2019; Peng et al., 2019; Rad and Lepetit, 2017). Such models are weighty and hence not always suitable for real-time platform deployment. Models such as the PoseCNN (Xiang et al., 2017) and T6D-direct (Amini et al., 2021) are able to regress the 6D poses, however, a very large dataset is required to train those models for better accuracy since they have no refinement module to count on. Pose estimation using depth modality often involve the conversion of depth image to point cloud and proceeds with the segmentation of object masks (Gao et al., 2021, 2020; Liu et al., 2022) adopted semantic segmentation from depth images and point clouds to regress 6D poses. This is accompanied by computational burden due to the conversion to point cloud and often requires a large dataset. In contrast, we utilised the raw depth modality for the regressed pose refinement without converting to point cloud as presented further in this paper.

A problem that can be envisaged in using models for pose estimation is the availability of a robust dataset. Thus, on the pose estimation dataset, We will discuss the related works under the following categories:

Fruit Datasets. The most used fruit dataset for classification problems is the Fruits 360 (Mureşan and Oltean, 2017). The dataset consists of 90,483 RGB images of different varieties of fruits. A Logitech C920 camera was used to capture the fruits on a white background.

Even though the dataset is rich in terms of the fruit variety, the dataset fails to capture multiple modalities thereby restricting the dataset to only classification and detection problems. Hence, 6D-pose estimation is not possible using this dataset. Additionally, images are of size 100×100 thereby making it difficult to utilize the data for very detailed classification problems. Additional modality becomes a necessity since the RGB camera can not provide the optimal dataset for all the listed applications. More works in this regard were carried out with the RGB-D setups (Lehnert et al., 2017; Tian et al., 2019; Tu et al., 2018; Wang et al., 2017) while others exploited the combination of RGB with Near Infrared (NIR) images (Gené-Mola et al., 2019; Sa et al., 2016). Even though these works have added an additional modality, the problem of 3D fruit localisation of the detected fruit still persists since the 6D poses of the camera and fruits are not readily provided as ground truths. Moreover, they focus on a single fruit as a class hence reducing the diversity of these data.

6D Pose Datasets. The LineMOD Dataset (Hinterstoisser et al., 2013) is a well-known dataset for 6D pose estimation. It consists of RGB-D images and poses of 15 objects. The OCCLUSION dataset (Brachmann et al., 2014) has properties like the former but also accounts for testing 6D poses of occluded objects. Another dataset with similar property is the T-LESS Dataset (Hodan et al., 2017). This dataset is accompanied by 3D card models. YCB-Video Dataset (Xiang et al., 2017), a popular dataset for 6D pose estimation consists of household objects displayed in short videos. Other 6D pose estimation datasets are also proposed (Doumanoglou et al., 2016b; Tejani et al., 2014; Xie et al., 2013). Although these datasets are tailored for 6D-pose estimation, none of them is targeted towards the 6D-pose estimation of fruits to facilitate fruit picking application in precision agriculture.

Other Multi-Modal Datasets. The KITTI dataset (Geiger et al., 2013) is a popular multi-modal dataset having data from lidar, stereo and IMU sensors. Due to the richness of this dataset, it has facilitated the emergence of state-of-art methods for 3D-object detection. The H3D dataset (Patil et al., 2019) is another multi-modal dataset where objects are annotated from multiple views as opposed to KITTI. Other significant multi-modal datasets have been proposed over the years (Choi et al., 2018; Geyer et al., 2020; Houston et al., 2021; Sun et al., 2020; Zhu et al., 2020). However, the majority of the datasets are proposed for autonomous driving and can barely find applications in fruit detection, fruit 6D-pose estimation and fruit picking.

Platforms. In terms of platform, the majority of the indoor multi-modal datasets are acquired with handheld methods (Dai et al., 2021; Lee et al., 2019). This does not account for more dynamic movement and rich poses especially when applied to robotics platforms.

Summarily, the literature has shown that the early pose estimation methods suffer when the objects are textureless. To compensate for that, other methods proposed using geometric information to improve the prediction accuracy. However, these methods including using of local and global descriptors proved to be computationally expensive and poorly predicted the pose on reflective objects. Other challenges arise from other proposed methods. they include pose estimation of occluded objects, symmetry of objects, object truncation, multiple modalities and datasets.

5.3 TransPose Pipeline

TransPose architecture performs two interdependent tasks to obtain the final 6DoF pose of the target object. As seen in Fig. 5.1, an RGB image is used as the input to the pipeline.

The input image is passed to the transformer network which has a ResNet-101 (He et al., 2016a) backbone for features extraction. These features are then passed to the transformer model consisting of a standard encoder and decoder setup (Vaswani et al., 2017). The model is used to obtain an image patch by detecting the object and assigning a ROI to the detected object. Also, the transformer is used to regress the initial pose of the object in the frame. The second segment of the architecture is the depth estimation and refinement module. Upon the completion of the transformer stage, the pose refinement using the pose is initiated. The depth estimation network encompasses a feature pyramid network (FPN) (Lin et al., 2017) that

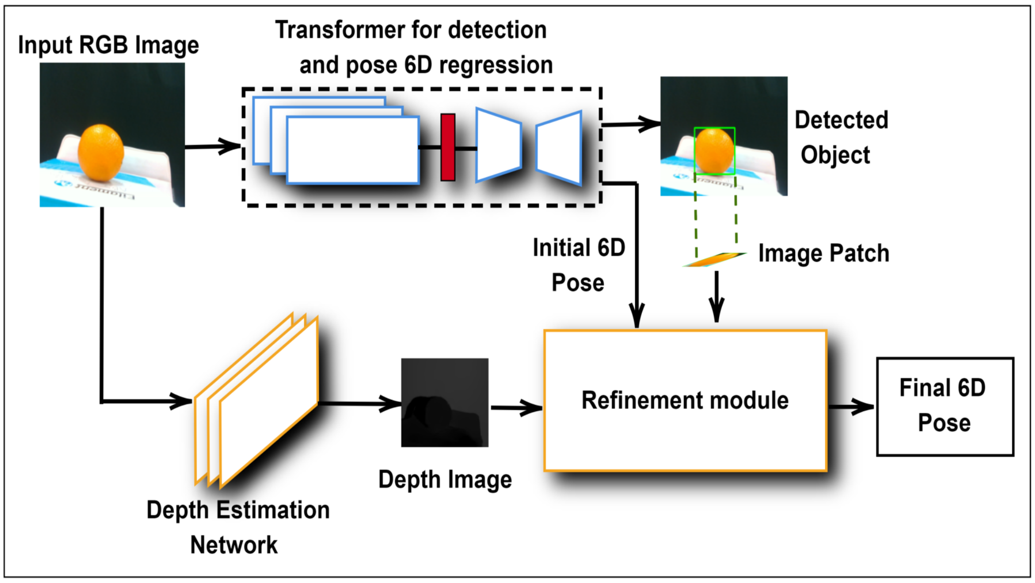


Fig. 5.1 Overall network architecture that performs object detection, depth prediction and 6D pose prediction.

takes in the same RGB image input as the transformer and gives an output of the estimated depth image. The image patch obtained from the transformer model is used to isolate the target on the depth image and hence obtain the depth of the target from the camera. The depth is then used to compute other components of the translation and subsequently used to refine the estimated 6D pose of the target. The following are the contributions proposed in the TransPose pipeline:

- a novel pipeline for 6DoF object pose prediction that favourably compares with other state-of-the-art methods
- As part of the pipeline, a lighter depth estimation network that utilizes a better up-sampling method for depth prediction is proposed.
- Additional analyses are conducted with our own generated fruit dataset to facilitate an evaluate 6D pose estimation performance for fruit picking applications.
- A first-ever novel fruit dataset with multiple modalities (thermal, depth, RGB) tailored specifically for pose estimation.

The pipeline for TransPose 6D object pose estimation can be divided into three main parts:

- Detection and Regression Transformer (DRT)
- Depth Estimation Network (DEN)
- Refinement module for final 6D pose estimation.

5.3.1 Detection and Regression Transformer

This transformer is mainly adopted for object detection, image patch designation and initial relative 6DoF pose regression. The transformer architecture is inspired by Detection Transformer DETR (Carion et al., 2020) and T6D-Direct (Amini et al., 2021). The model is represented in Fig. 5.2.

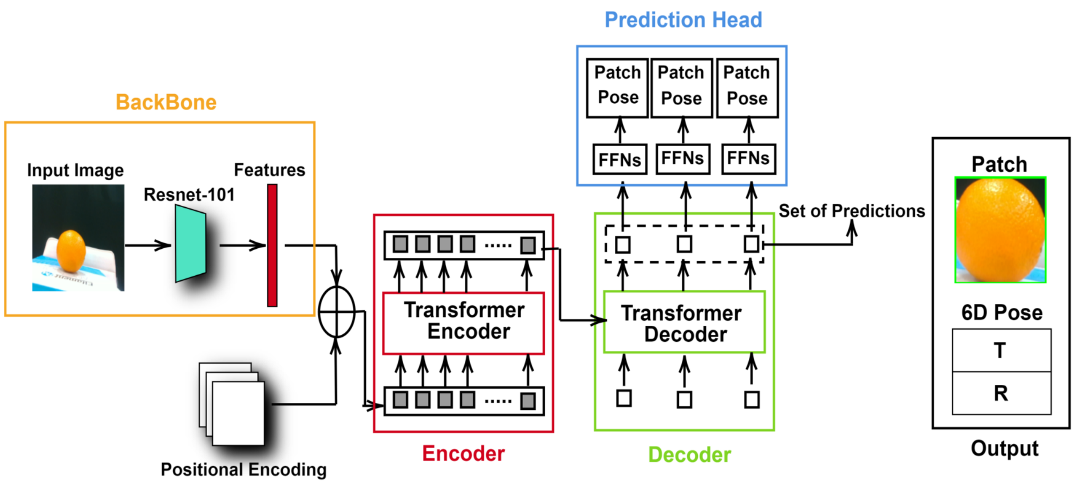


Fig. 5.2 Transformer for detection, image patch and initial 6D pose regression.

An RGB image is used as the input of the model. A ResNet-101 is used as the CNN backbone to extract features and create a vector which is used as an input to the transformer encoder-decoder. Sets of predictions of size N_c are produced by the transformer encoder-decoder. Prediction heads are added in the form of Feed Foward Networks (FFN) to regress the pose and patch. We can further categorise the losses of the model as follows:

Set Prediction Loss

The patch prediction in the form of ROI is obtained by assigning a bounding box around the object of interest. From the input image through the decoder, the model produces a set of size N_c of tuples with fixed cardinality, where N_c also corresponds to the maximum number of the expected targets within the image. The content of each tuple is the image patch (left bottom pixel coordinates, height and width), class label probabilities and 6D pose (translation and rotation) of the predicted object. A bipartite matching is adopted to match the ground truth and the predicted sets to obtain matching pairs. The model is then trained to minimise a loss between the pairs.

Consider ground truth objects x_1, x_2, x_3, ... x_n, let's assume N_c is more than the number of objects in the image, bipartite matching is performed to match the ground truth x which is a set of size N_c padded with no-object ($\emptyset$) with the predicted set $\hat{x}$ of the same size. Essentially, performing a permutation between the sets while minimizing the loss below.

$$\hat{\rho} = \underset{\rho \in \Theta_{N_c}}{\arg\min} \sum_{i}^{N_c} \mathscr{L}_{match}(x_i, \hat{x}_{\rho(i)}) \tag{5.1}$$

$\mathscr{L}_{match}(x_i, \hat{x}_{\rho(i)})$ is the pair-wise match cost between the prediction at index $\rho(i)$ and the ground truth tuple x_i.

Hungarian loss

After matching, the model is trained to minimise the Hungarian loss. The predicted patch is denoted as $\hat{\gamma}_{\rho(i)}$. Thus, the Hungarian loss is defined as (Amini et al., 2021):

$$\begin{aligned} \mathscr{L}_{hung}(x_i, \hat{x}) = & \sum_{i}^{N_c} [\lambda_{pose} \mathbb{1}_{c_i \neq \emptyset} \mathscr{L}_{pose}(R_i, t_i, \hat{R}_{\hat{\rho}(i)}, \hat{t}_{\hat{\rho}(i)}) \\ & - log \hat{P}_{\rho(i)}(c_i) + \mathbb{1}_{c_i \neq \emptyset} \mathscr{L}_{patch}(\gamma_i, \hat{\gamma}_{\hat{\rho}(i)})] \end{aligned} \tag{5.2}$$

$\hat{\rho}$ is the lowest cost from Eq.5.1, c_i is the class probability and γ_i is a vector that defines the ground truth image patch coordinates, height and width.

Patch loss

The patch loss is a component of Eq. 5.2. The loss $\mathscr{L}_{patch}(\gamma_i, \hat{\gamma}_{\rho(i)})$ which combines l_1 loss and generalized IOU (Rezatofighi et al., 2019) is defined as follows:

$$\mathscr{L}_{patch}(\gamma_i, \hat{\gamma}_{\rho(i)}) = \sigma_1 \mathscr{L}_{iou}(\gamma_i, \hat{\gamma}_{\rho(i)}) + \sigma_2 ||\gamma_i - \hat{\gamma}_{\rho(i)}|| \tag{5.3}$$

and,

$$\mathscr{L}_{iou}(\gamma_i, \hat{\gamma}_{\rho(i)}) = 1 - \left(\frac{|(\gamma_i \cap \hat{\gamma}_{\rho(i)}|}{|(\gamma_i \cup \hat{\gamma}_{\rho(i)}|} - \frac{|L(\gamma_i, \hat{\gamma}_{\rho(i)}) \backslash \gamma_i \cup \hat{\gamma}_{\rho(i)}|}{|L(\gamma_i, \hat{\gamma}_{\rho(i)})|} \right) \tag{5.4}$$

$\sigma_1, \sigma_2 \in \mathbb{R}$ are hyperprameters. $L(\gamma_i, \hat{\gamma}_{\rho(i)})$ is the largest patch having the ground truth γ_i and the predicted $\hat{\gamma}_{\rho(i)}$.

$\mathscr{L}_{pose}(R_i, t_i, \hat{R}_{\hat{\rho}(i)}, \hat{t}_{\hat{\rho}(i)})$ is the pose loss. The pose loss is subdivided into two components, translation t and the Rotation R inspired by T6D-direct (Amini et al., 2021). conventional l_2 loss is used to supervise the translation while ShapeMatch loss (Xiang et al., 2017) is used for the rotation to cater for symmetrical objects.

$$\mathscr{L}_{pose}(R_i, t_i, \hat{R}_{\rho(i)}, \hat{t}_{\rho(i)}) = L_R(R_i, \hat{R}_{\rho(i)}) + ||t_i - \hat{t}_{\rho(i)}|| \tag{5.5}$$

$$L_R = \begin{cases} \frac{1}{|K|} \sum\limits_{y_1 \in K} \min\limits_{y_2 \in K} ||(R_i y_1 - \hat{R}_{\rho(i)} y_2)|| & \text{if symmertic,} \\ \\ \frac{1}{|K|} \sum\limits_{y \in K} ||(R_i y - \hat{R}_{\rho(i)} y)|| & \text{otherwise.} \end{cases} \tag{5.6}$$

K represents the 3D points set. $R_i$and t_i are the ground truth rotation and translation respectively. While $\hat{R}_{\rho(i)}$ and $\hat{t}_{\rho(i)}$ are the predicted rotation and translation respectively.

5.3.2 Depth Estimation Network

The depth network is inspired by the FPN architecture (Lin et al., 2017). The motivation is that FPNs are capable of extracting features at different scales. We adopt ResNet-101 as a backbone for feature extraction, two 3×3 convolutional layers to process features and ReLU as an activation function for the layers as seen in Fig. 5.3. Additionally, a better lightweight upsampling technique (Wang et al., 2019c) that covers a larger field of view and enables the generation of adaptive kernels for better prediction is utilised. The depth images are one-fourth of the original image's size. The gradient of the depth map is obtained using a Sobel filter. The depth loss adopted in the training of this network is l_1 norm loss defined as follows:

$$\mathscr{L}_{depth} = \frac{1}{n}\sum_{i=1}^{n}||d_i - \hat{d}_{(i)}|| \tag{5.7}$$

where, d_i and $\hat{d}_{(i)}$ are the ground truth depth and the predicted depth respectively.

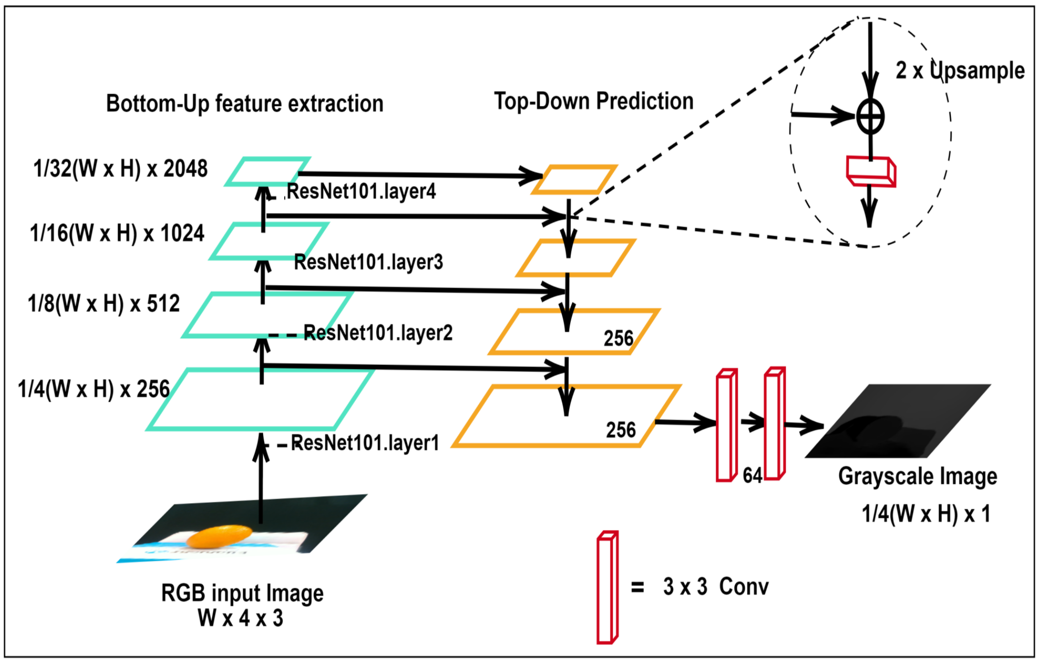


Fig. 5.3 Proposed Depth estimation network using FPN

5.3.3 Refinement module for final 6DoF pose estimation.

The refinement module consists of the depth patch generation and final pose estimation processes. The patch and the regressed 6D pose from the transformer alongside the depth image are used as inputs for the refinement module as shown in Fig. 5.4.

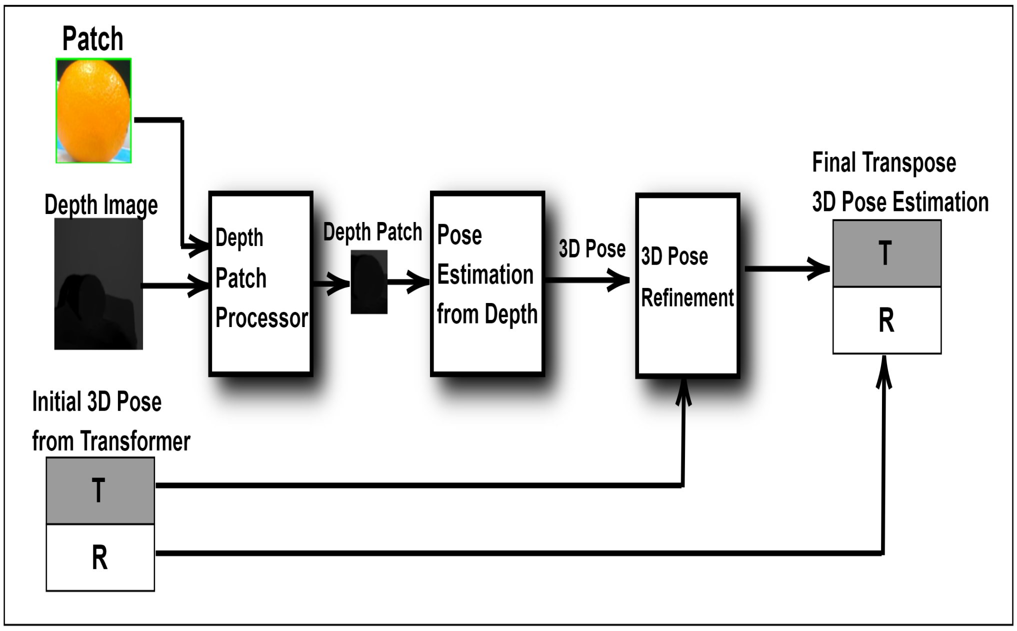

Patch
Depth Image
Depth Patch Processor
Depth Patch
Pose Estimation from Depth
3D Pose
3D Pose Refinement
Final Transpose 3D Pose Estimation
T
R
Initial 3D Pose from Transformer
T
R

Fig. 5.4 Refinement module for final 6D pose estimation

The Patch which is the ROI obtained from the transformer is formulated as:

$$\psi_i = [B_{opx}, B_{opy}, H_{op}, W_{op}] \tag{5.8}$$

where B_{px}, B_{py} represent the bottom left corner pixel coordinates of the patch respectively and H_p, W_p are the Height and Width of the patch respectively, all with respect to the original RGB image size $S_o = (W_o \times H_o)$. Let's represent the size of the depth image also as $S_d = (W_d \times H_d)$. where $S_o \neq S_d$. Thus we can obtain our depth patch ψ_j with respect to S_d from Eq. 5.8 as:

$$\psi_j = [B_{dpx}, B_{dpy}, H_{dp}, W_{dp}]$$

$$= \psi_i \times \begin{bmatrix} \frac{W_d}{W_o} & 0 & 0 & 0 \\ 0 & \frac{H_d}{H_o} & 0 & 0 \\ 0 & 0 & \frac{H_d}{H_o} & 0 \\ 0 & 0 & 0 & \frac{W_d}{W_o} \end{bmatrix} \tag{5.9}$$

where B_{dpx}, B_{dpy} represent the bottom left pixel coordinates of the depth patch respectively and H_{dp}, W_{dp} are the Height and Width of the depth patch respectively, all with respect to the depth image size S_d. The depth patch now represents our depth ROI which is the object in the depth image frame and thus we can obtain the depth t_{z1} from the camera to the target to be the depth information at the centre pixel of the depth patch, The centre pixel coordinate $C_d = (C_{dx}, C_{dy})^T$ can be obtained as follows:

$$\begin{aligned} C_{dx} &= B_{dpx} + \frac{W_{dp}}{2} \\ C_{dy} &= B_{dpy} + \frac{H_{dp}}{2} \end{aligned} \tag{5.10}$$

The translation from the depth network model t_1 utilises t_{z1} (which in this case is the depth from the camera and can be seen as the translation in the z-axis) to compute t_{x1} and t_{y1} which are the translations in x and y axis to complete the translation (computed from depth network) $t_1 = (t_{x1}, t_{y1}, t_{z1})^T$. Assuming the camera matrix is known, t_{x1} and t_{y1} can be obtained following the projection equation (see Section 2.1) of a pinhole camera model as follows:

$$\begin{bmatrix} C_{ox} \\ C_{oy} \end{bmatrix} = \begin{bmatrix} f_x \frac{t_{x1}}{t_{z1}} + P_x \\ f_y \frac{t_{y1}}{t_{z1}} + P_y \end{bmatrix} \tag{5.11}$$

where f_x and f_y represent the focal length of the camera, $(P_x, P_y)^T$ is the principal point. $C_o = (C_{ox}, C_{oy})^T$ is the centroid of the object which can be obtained from the image patch similar to Eq. 5.10 to be $(B_{opx} + \frac{W_{op}}{2}, B_{opy} + \frac{H_{op}}{2})^T$ assuming the centroid coincides with the centre of the patch. Finally t_{x1} and t_{y1} can be obtained as:

$$\begin{bmatrix} t_{x1} \\ t_{y1} \end{bmatrix} = \begin{bmatrix} \frac{(C_{ox} - PP_x) t_{z1}}{f_x} \\ \frac{(C_{oy} - PP_y) t_{z1}}{f_y} \end{bmatrix} \tag{5.12}$$

$$t_1 = (t_{x1}, t_{y1}, t_{z1})^T \tag{5.13}$$

Finally, we can obtain the final fusion-based object translation t as:

$$t = (w_1 \times t_1) + (w_2 \times t_2) \tag{5.14}$$

where the weights $w_1, w_2 \geq 0$ such that $w_1 + w_2 = 1$. t_1 is the computed translation from the depth in equation. 5.13 and t_2 is the regressed translation from the transformer model. Note that w_1 and w_2 are selected depending on the performance of both the transformer and depth model. The model with a lower loss will have a higher w and vice-versa.

5.4 Fruity Dataset

As part of this chapter, it is required to have a rich dataset so that the TransPose model can be effectively trained. Unfortunately, the availability of datasets in the agricultural domain is quite low compared to other domains like autonomous driving. Although the application of robotic platforms for precision agriculture is gaining traction in modern research, the demand for a complete fruit dataset is still not satisfied. Thus, the Fruity dataset is proposed. Fruity is a multi-modal fruit dataset with a variety of use cases such as 6D-pose estimation, fruit detection, fruit picking applications, etc. As far as we know at the time of writing this thesis, this dataset is the first-ever multi-modal fruit dataset tailored specifically for fruit 6D pose estimation in precision agriculture. The dataset is collected over a range of multiple sensors consisting of an RGB-D camera, a thermal camera and an indoor tracking camera for ground truth poses. Fruity features RGB images, stereo depth images, thermal images, camera 6D-poses, fruit 6D-poses and relative 6D-poses between the cameras and fruits. The classes of the dataset are commonly harvested fruits which include: apples, oranges, bananas, avocados and lemons. It is also enriched with a clustered class to account for occlusion scenarios. The dataset is recorded over multiple trajectories implemented with multiple platforms encompassing a robotic manipulator and an Unmanned Aerial Vehicle (UAV).

5.4.1 Motivation for the Fruit Dataset

Precision agriculture is poised to be the solution to the global food shortage. Robotics in agriculture is often considered to be a good form of precision agriculture. However, the shortage of accurate and complete datasets is restricting the exploitation of robots in agriculture. Detection and estimation of 6DoF poses find application in object grasping, Virtual Reality (VR), Augmented Reality (AR), and autonomous driving. However, the availability of datasets has limited the exploitation of such tools for agriculture. Having a single modality dataset introduces limitations that are associated with the sensing mechanism. For example, RGB cameras are not usable for complex computer vision applications in low illumination scenarios (Li et al., 2022). RGB Cameras are utilised for edge detection, colour-based classifications and 2D localisation. However, using these images for 3D localisation can be a challenging task (Caesar et al., 2020). Understanding this compromise, researchers complement the weakness of one sensor with the strength of another thereby arriving at the concept of multimodal sensoring. Even though datasets are collected peculiar to a given application, it is pertinent that they are collected with completeness, accuracy, and richness to facilitate the development and evaluation of newer innovations.

Over the years, many datasets have been collected for autonomous driving (Caesar et al., 2020), pedestrian detection (Cong et al., 2022) and odometry (Li et al., 2022). However, very few have been collected for fruit detection and picking. Thus, Fruity: A multi-modal dataset for fruit recognition and 6D-Pose Estimation in precision agriculture is proposed. This dataset can be utilized to facilitate and evaluate new innovative methods in fruit picking, detection, and 3D localization. Fruity consists of 6 classes namely: apple, banana, orange, avocado, lemon, and a fruit cluster class. The modalities of the dataset include a thermal modality, RGB modality and a depth modality as seen in Fig. 5.5. Each class of the dataset is accompanied by 6D poses of the fruits and the camera alongside the relative poses between them. The dataset is acquired through different trajectories on multiple platforms. A customised sensor rig is designed and constructed to house the sensors while being mounted on the platforms. The data acquisition and synchronisation are possible through the ROS framework (Quigley et al., 2009) as shown in Fig. 5.6.

The outputs of the system are the RGB, depth, and thermal images. The 6DoF poses of the cameras and the targets are also obtained from the system. These poses are also used to compute the relative poses between the camera and the target. The major contributions of this dataset can be summarised as follows:

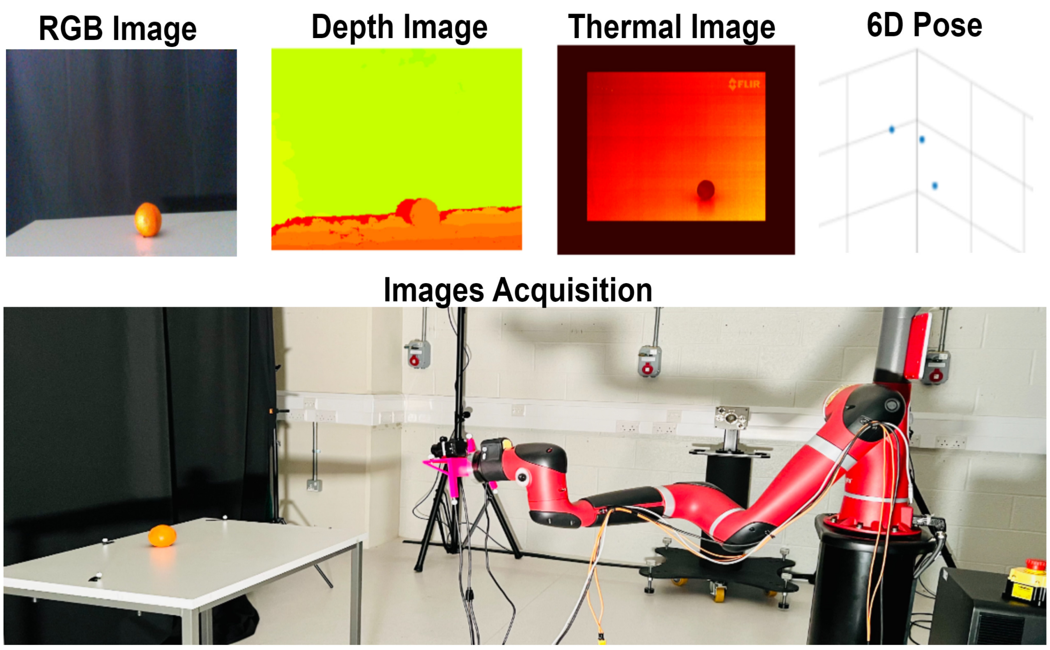


Fig. 5.5 Figure showing the modalities of our dataset (RGB, Depth and Thermal) including the 6D Pose and the robotic manipulator used for acquiring the dataset.

- A multi-modal indoor fruit dataset that encompasses data from modern sensors is proposed. This is the first multi-modal fruit dataset that is tailored for 6D-pose estimation in precision agriculture which finds application in autonomous fruit picking and harvesting.
- The dataset is acquired over different trajectories implemented on multiple platforms (manipulator and UAV) to provide a variety of 6D poses to facilitate effective training.
- A toolkit to easily manage and utilize the multi-modal dataset in the form of plug-and-play codes as well as providing documentation on how to easily use this data for the agriculture research community.

5.4.2 Acquisition Process

The acquisition process can be grouped into the following:

- Sensing hardware

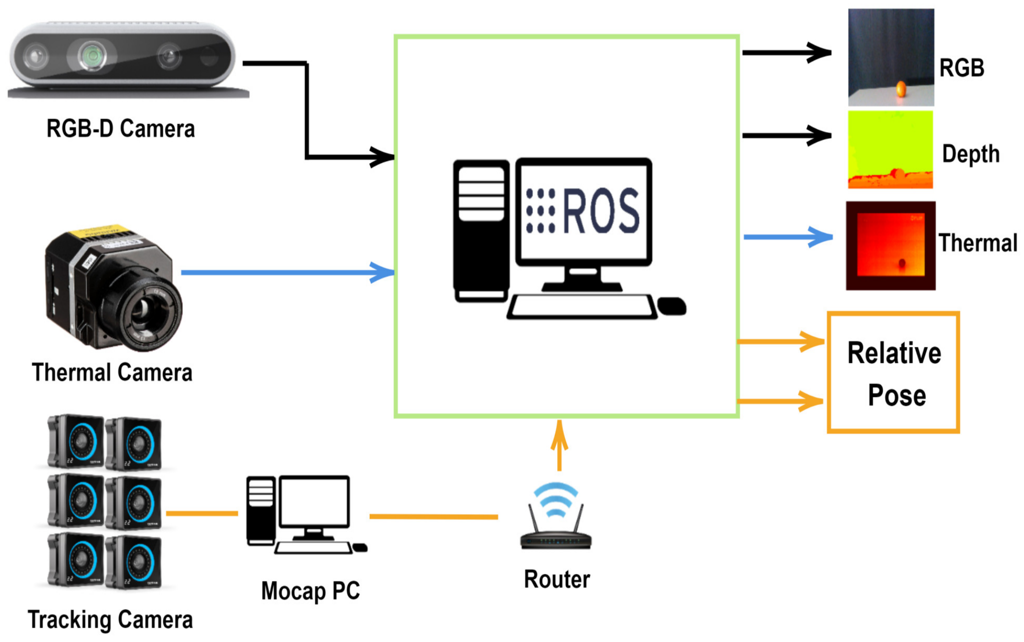


Fig. 5.6 Figure showing the overview of the dataset acquisition pipeline featuring RGB-Camera, Thermal Camera and Tracking system to record data simultaneously via ROS

- Sensor interfacing
- Platforms

Sensing hardware

To collect data from moving platforms, it becomes necessary to house the sensors on a befitting sensor rig to allow seamless data acquisition. We designed a CAD model of a sensor rig peculiar to our application. The sensor rig is capable of carrying all the required cameras (RGB-D and thermal) as seen in Fig. 5.7. It also has reflective markers onboard to allow for tracking and retrieval of its ground truth 6D pose. The sensors used for the data capture include an Intel RealSense D435i stereo camera, a FLIR vue pro thermal camera and an optitrack tracking system. The expected outputs from these sensors are summarised in Table 5.1. The stereo camera provides the RGB image and the depth image. The thermal camera provides the thermal information of the fruit and the tracking cameras are used to provide the 6DoF positions of the camera and the fruit.

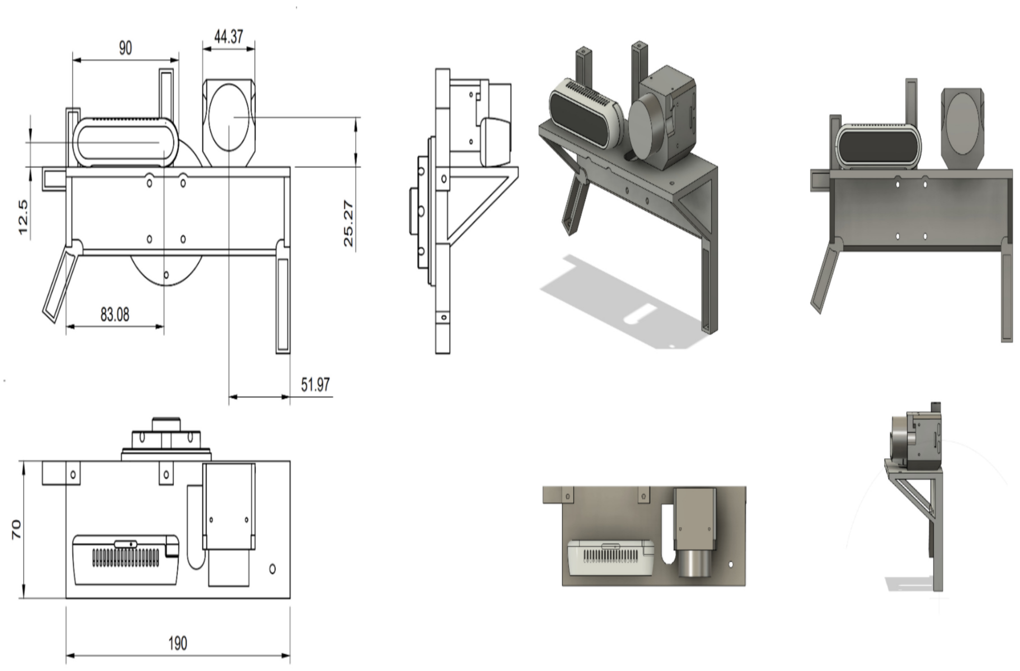


Fig. 5.7 Sensor Rig design drawing and CAD model

Table 5.1 Sensor specifications, types and outputs

-	1	2	3
Sensors	1 × RGBD Camera	1 × Thermal Camera	6 × Tracking Camera
Type	Intel real sense D435i stereo camera	FLIR Vue Pro thermal camera	Optitrack motion tracking system
Platform Used	UAV, Manipulator, Handheld	UAV, Manipulator, Handheld	N/A
Output	30Hz 8bit 640×480 RGB image 30Hz 16bit 640×480 depth image	30Hz 16bit 640×480 Thermal image -	3-dimensional translation 3-dimensional rotation

Sensor interfacing

ROS is used for the sensor interfacing. For a consistent dataset, we require all the data to be synchronous and in real time. This is implemented by collecting all the data from the sensors simultaneously as rostopics. Fig. 5.8 shows the interfacing of the sensors. The RGB-D camera is connected to the workstation and accessed through the real sense camera package.

This package collects the RGB and the depth data which are then published as rostopics. The data is then subscribed and synchronised before outputting both images. The same pipeline applies for the thermal camera but in this case, using a thermal camera package. The tracking cameras are hosted on another PC running the Optitrack software. The 6D poses are collected through a client package over wireless communication.

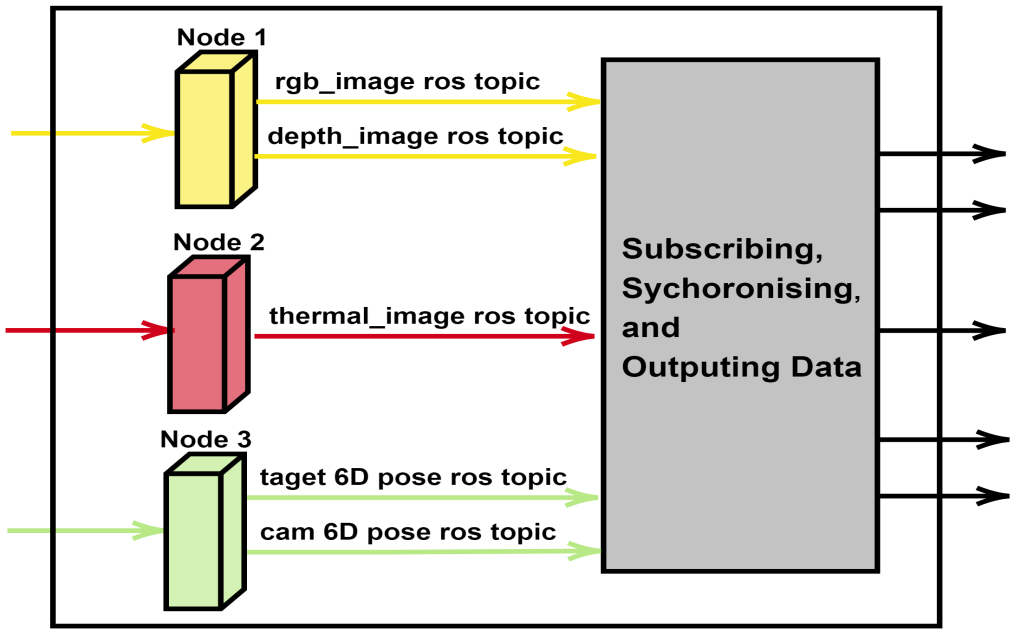


Fig. 5.8 ROS workflow with Node 1,2 and 3 representing the camera packages of our sensors. Respective data are published in the form of ROS topics which are then utilised in a script to synchronise and save the data

5.4.3 Platforms

The platforms used for the dataset collection are the Sawyer manipulator from Rethink robotics, and a customized UAV shown in Fig. 5.9. The camera rig is also held at hand to collect more data which is otherwise difficult to obtain from both platforms. Additionally, the technique enriches the data with various forms of movement thereby constituting more dynamic relative poses between the cameras and the fruit.

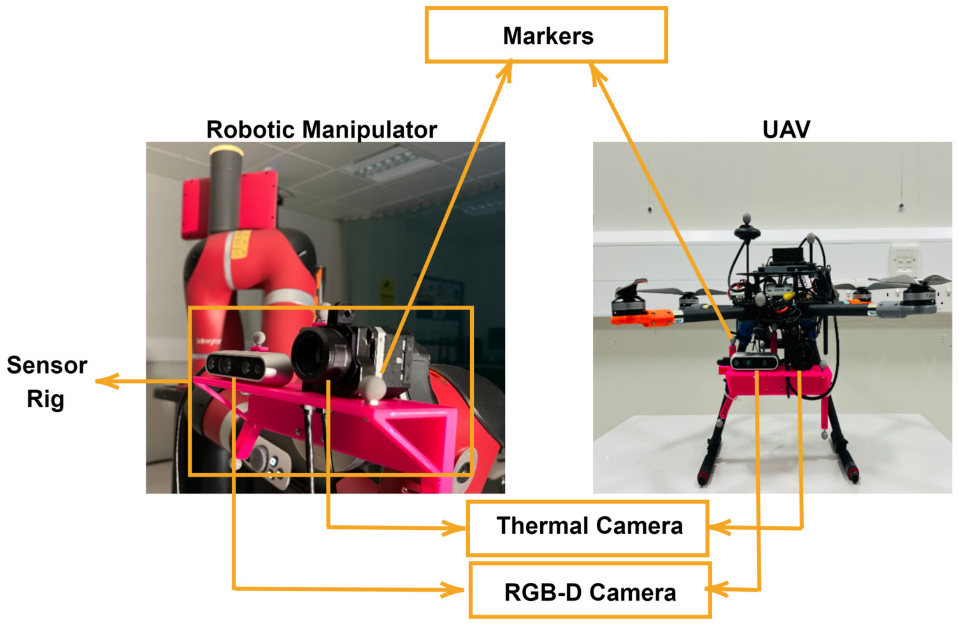


Fig. 5.9 Figure showing the platforms used to implement the trajectories for data acquisition. The Robotic Manipulator and UAV platforms are equipped with an RGB-D and thermal camera which are housed in the sensor rig

The Dataset Collection Process

The images and 6D poses of the fruits are collected sequentially in the form of trajectories. The platforms equipped with the required sensors are subjected to different trajectories. This gives us a rich dataset with a variety of poses as there are many different relative poses between the cameras and the fruits along each trajectory. The trajectories are determined by the cameras' FOV, and the freedom of the platforms' joints. A definitive trajectory is likely to have frames with no objects in sight which will not only confuse the network to be trained but also add more frivolous volume to the dataset. Thus, the trajectories are intuitively implemented to cater for these limitations. Frames are captured at 30Hz along the trajectory while simultaneously recording the relative 6DoF pose. The 6DoF pose P is represented as follows:

$$P = [X_t, Y_t, Z_t, X_r, Y_r, Z_r, W_r]^T \tag{5.15}$$

where X_t, Y_t, Z_t are the 3D-translation in X, Y, Z while X_r, Y_r, Z_r, W_r are the quaternions.

Collection on Manipulator

The Sawyer robotic manipulator is also used for the dataset collection. It has 7 degrees of freedom with a payload capacity of up to 4kg. The 3D-printed sensor rig carrying the camera setup is mounted as an end-effector to the manipulator. The trajectories are created as waypoints on the Intera software as seen in Fig. 5.10.

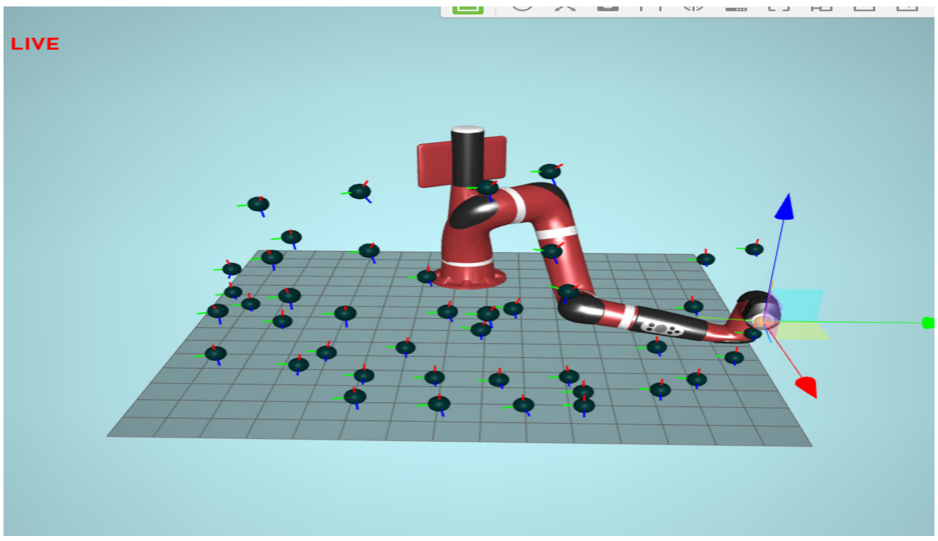


Fig. 5.10 Intera Software GUI showing waypoints

The trajectories are such that the FOV and the manipulator restraints are not impacted. A total of 5 trajectories are conducted with the manipulator. Fig. 5.11 shows the trajectories and the 3D-translation profile in the X, Y, Z direction of the manipulator. Fig. 5.12 shows the quaternions of the trajectories.

Collection on UAV

For more dynamic and rich poses, UAV is utilised to collect data from more distinct poses. The sensor rig is transferred to a UAV equipped with an onboard processing unit. The UAV is manually controlled to perform trajectories such that the target fruit remains in the FOV of our sensors. A total of 4 trajectories are conducted. Fig 5.13 shows the UAV's 3D-translation in X, Y, Z direction with quaternion shown in Fig 5.14.

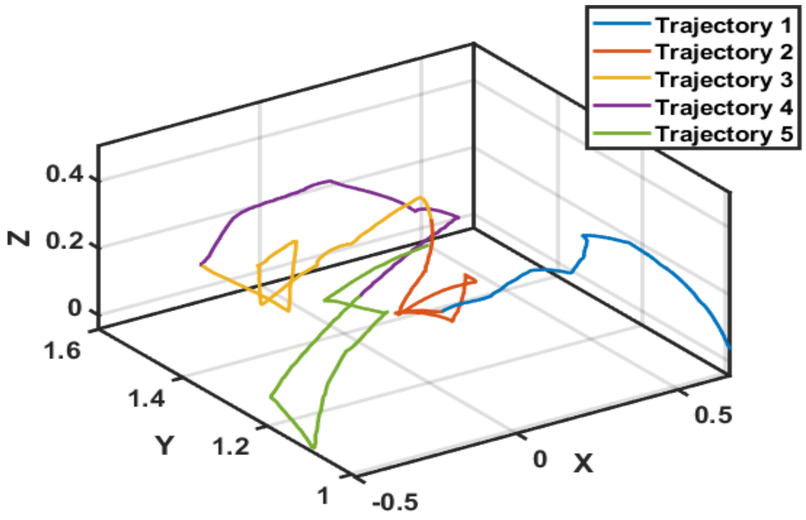


Fig. 5.11 Figure showing the 3D-translation $[X_t, Y_t, Z_t]$ in X, Y, Z direction of the manipulator's trajectories. Each trajectory is represented by a coloured line

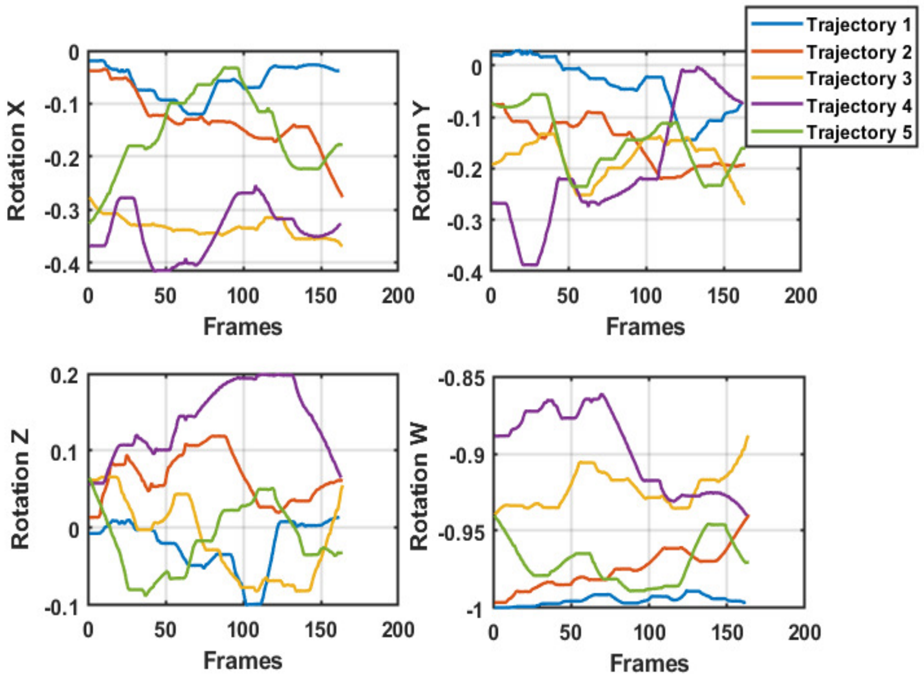


Fig. 5.12 Figure showing the quaternion profile $[X_r, Y_r, Z_r, W_r]$ of the manipulator's trajectories. Each trajectory is represented by a coloured line

Collection on Handheld Rig

The sensor rig is handheld to implement trajectories that are rather not possible on both the manipulator and the UAV to provide more coverage and multiple poses. The trajectories

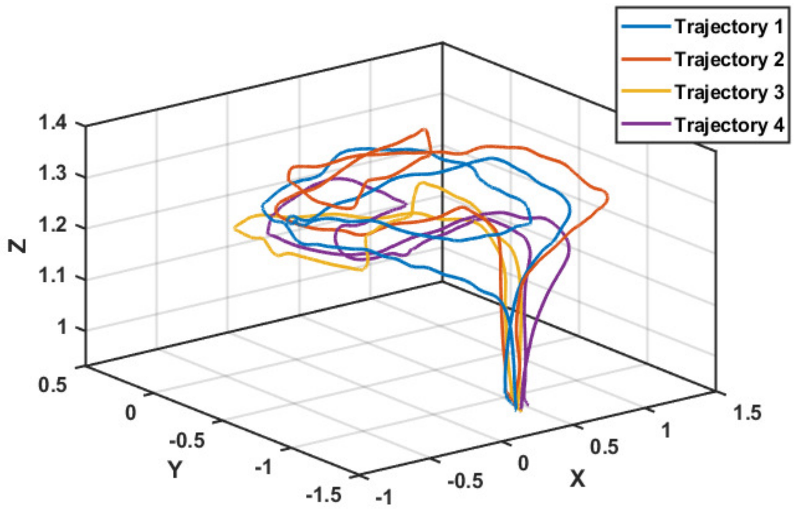


Fig. 5.13 Figure showing the 3D-translation $[X_t, Y_t, Z_t]$ in X, Y, Z direction of the UAV's trajectories. Each trajectory is represented by a coloured line

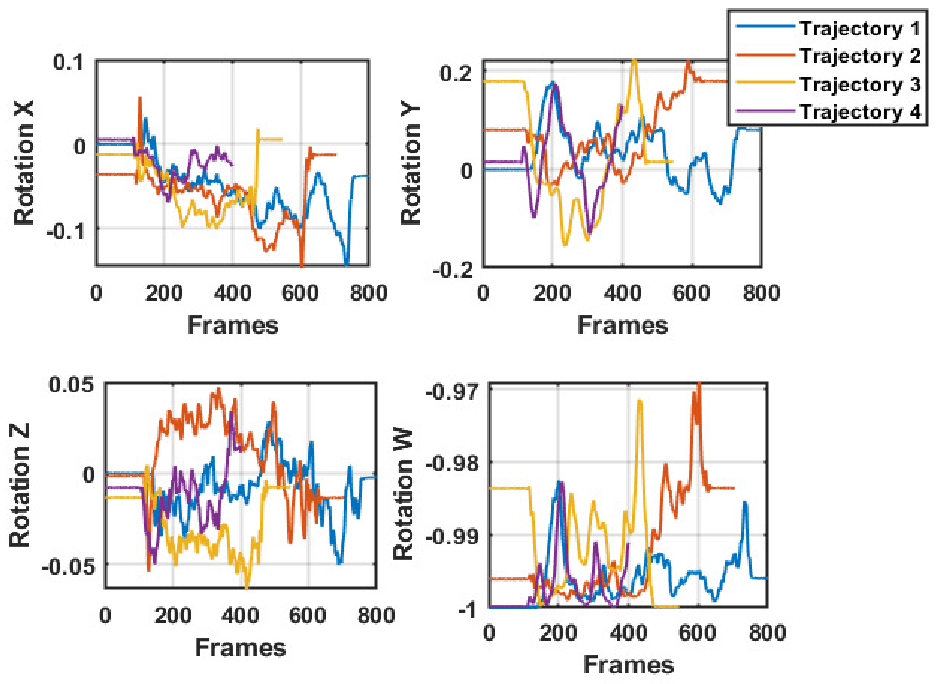


Fig. 5.14 Figure showing the quaternion profile $[X_r, Y_r, Z_r, W_r]$ of the UAV's trajectories. Each trajectory is represented by a coloured line

are implemented by randomly moving through a space to cover the maximum possible area without influencing the FOV. A total of 6 trajectories are conducted. As seen in Fig 5.15 and Fig. 5.16, more areas are covered thereby providing more 6D-pose.

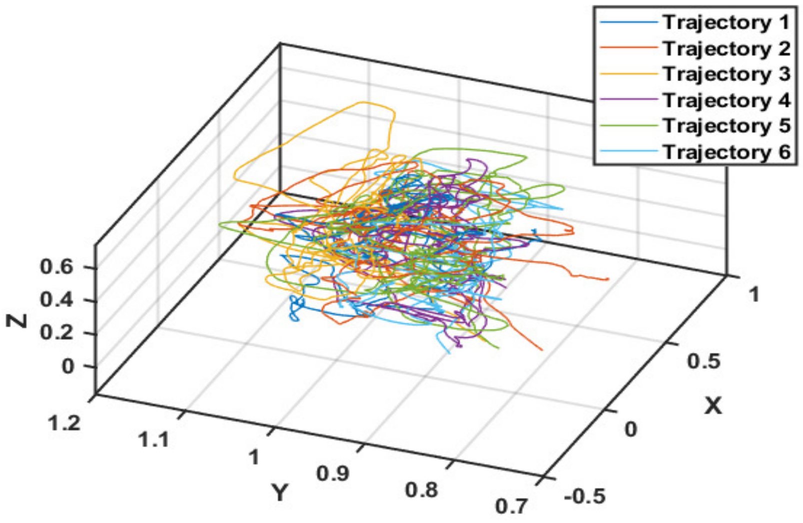


Fig. 5.15 Figure showing the 3D-translation $[X_t, Y_t, Z_t]$ in X, Y, Z direction of the handheld trajectories. Each trajectory is represented by a coloured line

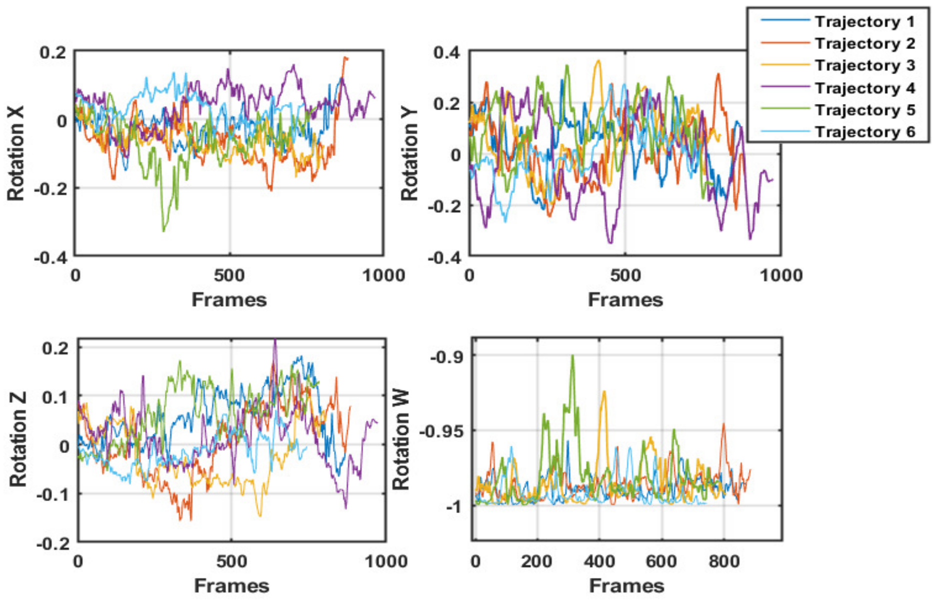


Fig. 5.16 Figure showing the quaternion profile $[X_r, Y_r, Z_r, W_r]$ of the handheld trajectories. Each trajectory is represented by a coloured line

5.4.4 Dataset Description and Distribution

A total of 31,195 images are collected which are distributed across the 3 modalities (RGB, depth, and thermal). The camera 6D-pose, fruit 6D-pose, and the relative 6DoF-pose between the camera and the fruit are also acquired. A cumulative total of 15 distinct trajectories are implemented on 3 platforms to offer a variety of poses and images. Fig 5.17 shows the distribution of the dataset. The apple class has a total of 1869 images for each modality while the avocado class has 1900 for each modality. The banana, lemon, orange and cluster classes have 1805, 1788, 1719, and 1984 images respectively for each modality. Each of the captured images is accompanied by a 6D pose of the cameras, fruit, and the relative pose between them.

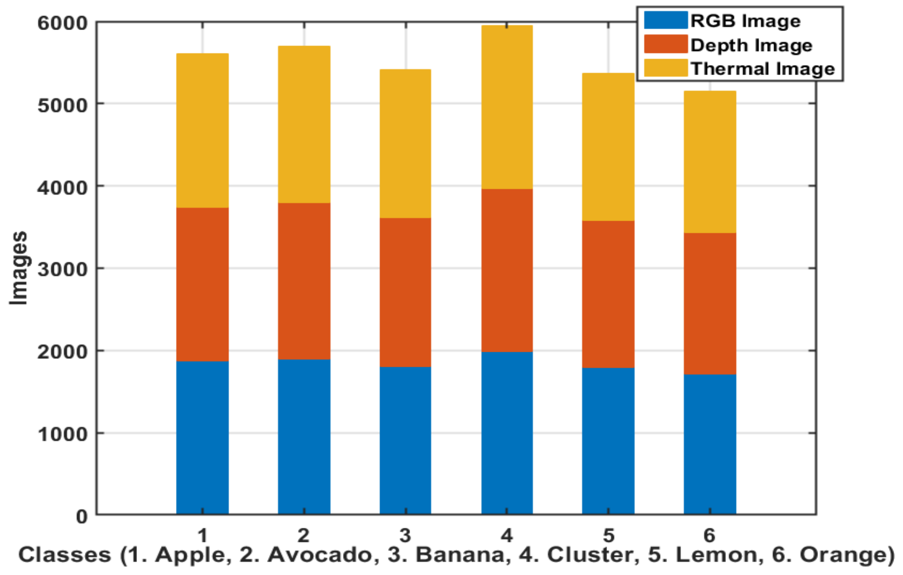


Fig. 5.17 Figure showing the distribution of the dataset, each colour represents a modality in the dataset

The dataset can be visualised in terms of distribution per class (apple, avocado, banana, lemon, orange and cluster) and distribution per modality (RGB, depth and thermal) as shown in Fig. 5.18 and 5.19 respectively. The general dataset distribution is shown in Fig. 5.20. It can be observed from the figures that the images acquired has a balanced distribution. This will facilitate effective training of models since any particular class will not be favoured due to having a higher contribution to the dataset.

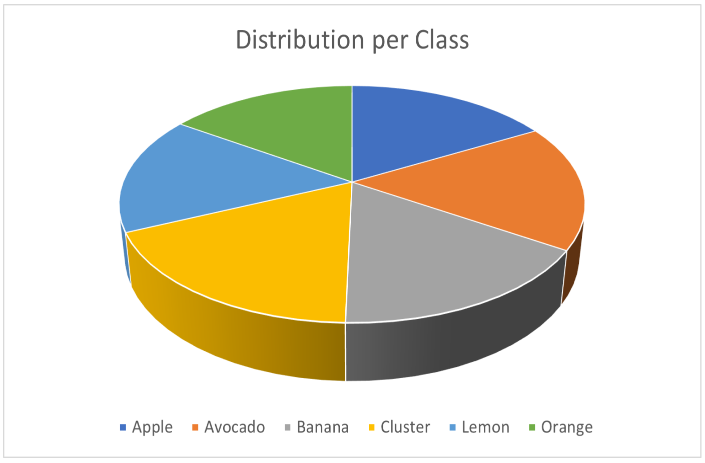


Fig. 5.18 Figure showing the distribution of the dataset, each colour represents a modality in the dataset

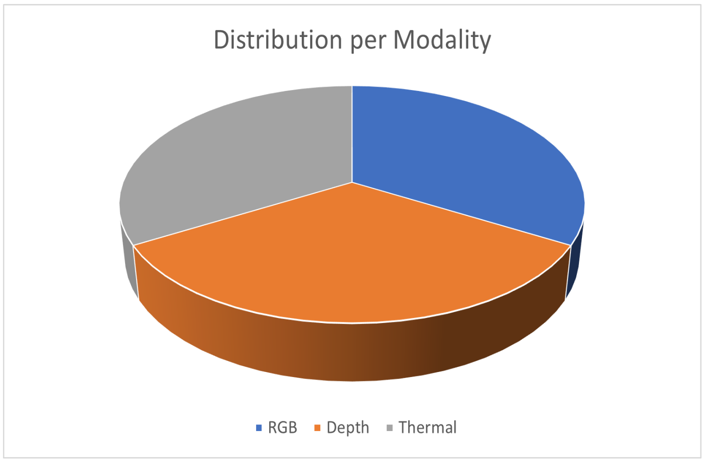


Fig. 5.19 Figure showing the distribution of the dataset, each colour represents a modality in the dataset

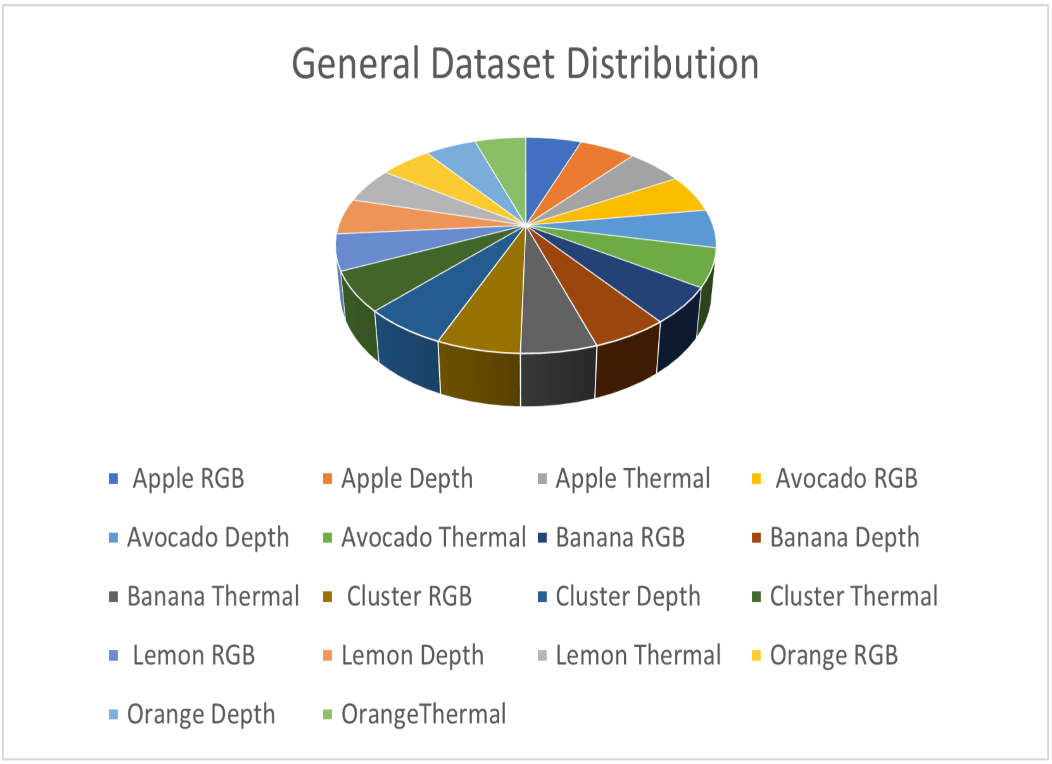


Fig. 5.20 Figure showing the distribution of the dataset, each colour represents a modality in the dataset

The Fruity dataset is compared with other fruit-related work/datasets available in Table 5.2. The dataset was compared based on modalities, classes, the platform used for dataset acquisition and 6D pose Ground Truth (GT). Our dataset proves to be a more complete dataset for fruit recognition and 6D-pose estimation.

Table 5.2 Comparison with other fruit datasets

Dataset	Platform	Class > 1	RGB	Depth	Thermal	6D-Pose
Fruit 360 (Mureșan and Oltean, 2017)	X	✓	✓	X	X	X
Sa et al. (Sa et al., 2016)	X	✓	✓	X	X	X
Kuang et al. (Kuang et al., 2018)	X	✓	✓	X	X	X
Tu et al. (Tu et al., 2018)	X	X	✓	✓	X	X
Gene et al (Gené-Mola et al., 2019)	X	X	✓	✓	X	X
Wang et. al (Wang et al., 2017)	X	X	✓	✓	X	X
Tian et. al (Tian et al., 2019)	X	X	✓	✓	X	X
lehnert et. al (Lehnert et al., 2017)	Manipulator	X	✓	✓	X	X
Ours	UAV Manipulator	✓	✓	✓	✓	✓

The qualitative result of the dataset is shown in Fig 5.21 for each modality captured at different 6D-poses. The depth and thermal image are displayed in different colourmaps to provide visual distinction between the modalities. However, the original images in the dataset are in grayscale as in most conventional datasets.

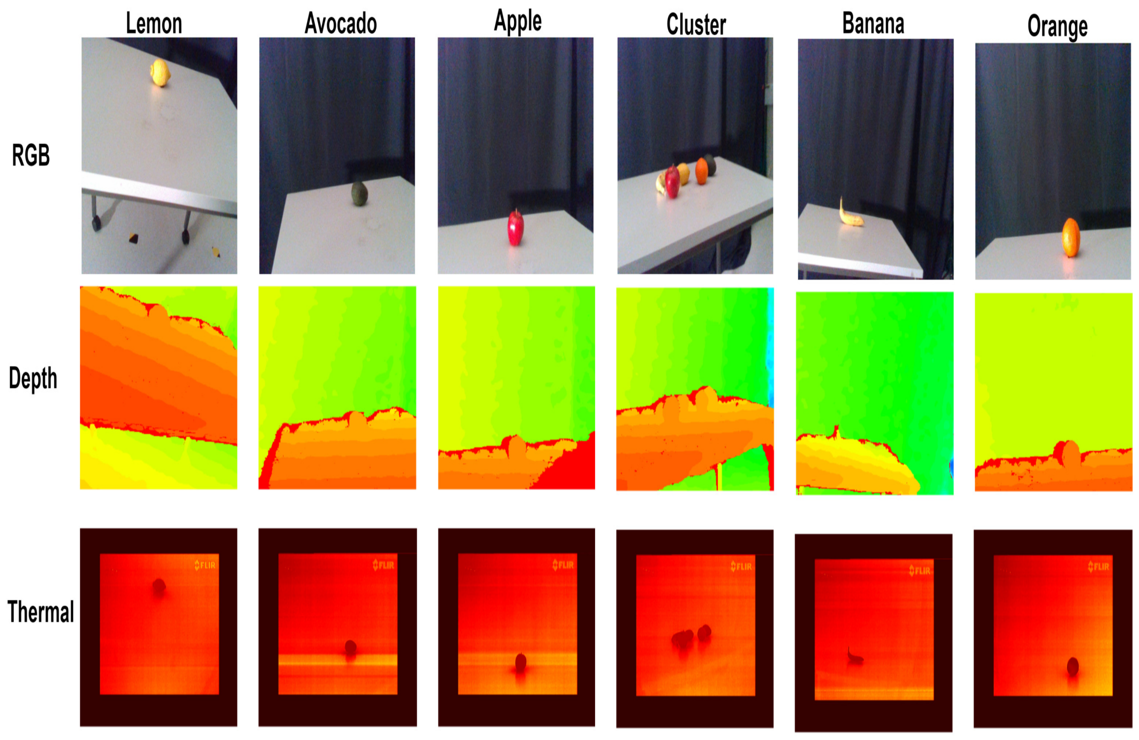


Fig. 5.21 Examples of Multi-Modal our dataset -top row from RGB-cameras, middle row from depth cameras and bottom row from thermal cameras The sensor rig was utilised to capture the individual fruit classes at different 6D-poses.

Remark 4 ***Fruity Dataset***

The Fruity dataset is acquired with a stereo camera, thermal camera, and tracking camera to provide more modalities of the target (fruits). The dataset is collected on different platforms in multiple trajectories to provide a variety of 6D poses to enrich the dataset. The multiple modalities can enhance the performance of neural networks in the detection and 6D pose estimation of fruits which can be applied to fruit picking. Owing to the increase in demand for robotics in agriculture, the aim is to provide the first fruit-based 6D pose estimation dataset to facilitate the use of artificial intelligence in precision agriculture. The custom dataset is utilised to train and test the TransPose pipeline for 6D pose estimation. Note that, the thermal modality was not utilised for this work. It only enriches the dataset for users that would use the dataset in the future for other related activities.

The popular YCB-Video dataset being a benchmark for 6D pose estimation (Xiang et al., 2017) is also used so results can easily be compared with other methods. The dataset has 133,936 images of 640 $\times$ 480 resolutions. Each image is accompanied with bounding boxes labels, depths, segmentation and 6D object pose annotations. Similar to (Xiang et al., 2017), test was carried out on 2,949 key frames from 12 scenes.

5.5 TransPose Evaluation Metrics

To evaluate the performance of the transpose model, some standard evaluation metrics are utilised. Firstly, recall that the model also encompasses a depth network. the depth estimation network is evaluated based on the *abs-rel*, *sq-rel*, *RMSE* and $RMSE_{log}$ proposed in (Eigen et al., 2014) as follows:

$$abs\_rel = \frac{1}{|T|}\sum_{i=1}^{T}\frac{|d_i - \hat{d}_i|}{\hat{d}_i} \tag{5.16}$$

$$sq\_rel = \frac{1}{|T|}\sum_{i=1}^{T}\frac{||d_i - \hat{d}_i||^2}{\hat{d}_i} \tag{5.17}$$

$$RMSE = \sqrt{\frac{1}{|T|}\sum_{i=1}||d_i - \hat{d}_i||^2} \tag{5.18}$$

$$RMSE_{log} = \sqrt{\frac{1}{|T|}\sum_{i=1}||\log d_i - \log \hat{d}_i||^2} \tag{5.19}$$

where T is the number of pixels in the test set.

For the overall pose estimation, the average distance (ADD) metric as suggested in (Pavlakos et al., 2017) is used for evaluation. This metric calculates the mean pairwise distance as follows:

$$ADD = \frac{1}{|K|}\sum_{y \in K}||(R_y + t) - (\hat{R}_y + \hat{t})|| \tag{5.20}$$

where R and t are the ground truth rotation and translation respectively. $\hat{R}$ and $\hat{t}$ are the predicted rotation and translation respectively. K is the set of 3D model points.

Average distance is calculated as the closest point distance for symmetrical objects as follows:

$$ADD - S = \frac{1}{|K|}\sum_{y_1 \in K}\min_{y_2 \in K}||(R_{y1} + t) - (\hat{R}_{y2} + \hat{t})|| \tag{5.21}$$

5.6 Training

The model is initialised as in (Carion et al., 2020) with pre-trained weights. The model utilises an input of image sized 640 $\times$ 480. The initial learning is set to 1.0^{-3} which is eventually decayed and the batch size is set to 16. AdamW optimizer (Loshchilov and Hutter, 2017) is used for training. The hyperparameters for calculating $\mathscr{L}_{patch}$ in Eq. 5.3, σ_1 and σ_2 are set to 2 and 5. Also, the parameter λ_{pose} for calculating $\mathscr{L}_{hungarian}$ in Eq. 5.2 is set to 0.05. The cardinality or number of prediction queries N_c is set to 21.

5.7 Experimental Results

The results are presented in the following categories:

- Depth estimation results
- TransPose results

5.7.1 Depth estimation results

For the depth estimation network, the training loss and accuracy per iteration are shown in Fig. 5.22. As the training proceeds, the training loss decreases thereby increasing the training accuracy per iteration.

The results obtained for the depth evaluation using the metrics in Eq. 5.16, 5.17, Eq. 5.18 and Eq. 5.19 are presented in Table. 5.3.

The performance of the proposed network is compared with other methods on the KITTI dataset and the custom fruit dataset. On the KITTI dataset, this method outperformed the others in the *sq-rel* and $RMSE_{log}$ metric and compares very closely with (Kuznietsov et al., 2017) in the *abs-rel* and *RMSE* metric. On the fruit dataset, this network outperforms the others in *abs-rel*, *sq-rel*, and $RMSE_{log}$ metric and compares closely in the *RMSE* metric. This comparison shows that it is sufficient to use this network for depth estimation as part of the TransPose pipeline. It can also be observed that as the training progresses, the training

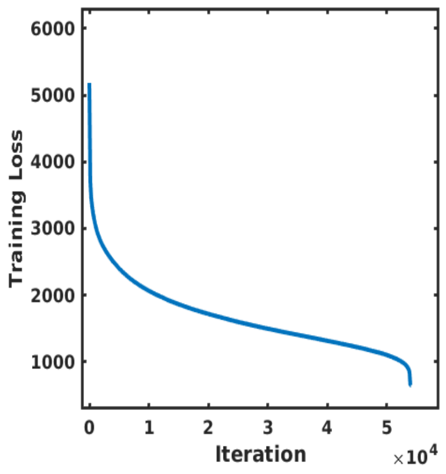


Fig. 5.22 Training loss and accuracy per iteration

Table 5.3 Depth estimation Network comparison with other methods

Method	Evaluation Metric (lower is better)			
	KITTI Dataset			
	abs-rel	*sq-rel*	*RMSE*	$RMSE_{log}$
Make3D (Saxena et al., 2008)	0.280	3.012	8.734	0.361
Eigen et al (Eigen et al., 2014)	0.190	1.515	7.156	0.270
Liu et al (Liu et al., 2015)	0.217	1.841	6.986	0.289
Kuznietsov et al (Kuznietsov et al., 2017)	**0.113**	0.741	**4.621**	0.189
Ours	0.114	**0.724**	4.694	**0.185**
	Custom Fruit Dataset			
Eigen et al (Eigen et al., 2014)	0.0885	1.3000	4.2440	0.2115
Liu et al (Liu et al., 2015)	0.0755	1.0917	3.9290	0.1938
Kuznietsov et al (Kuznietsov et al., 2017)	0.0499	0.5350	2.6907	0.1427
Ours	**0.0434**	**0.5153**	**2.5013**	**0.1342**

loss keeps declining while the accuracy appreciates. This signifies that the network is learning as the training proceeds. It is worth noting that we are not solemnly after a very high accuracy depth estimation network as this comes at a computational cost and the depth estimation network is just one part of the TransPose pipeline. Thus, a reasonable trade-off between computational cost and accuracy can be established to satisfy both decent estimation and future real-time implementation. Hence, the depth results are satisfactory for our purpose.

The qualitative result is shown in Fig. 5.23. Samples from all the classes encompassing their ground truths and the corresponding predictions are shown. A colour map is added to the depth images for visualisation.

Further comparison with other methods is carried out across each class of fruit. Fig 5.24 shows the comparison of each class of the fruit dataset using the *Abs-rel* and *sq-rel* metrics. From the results, the proposed depth network outperformed all the methods across all the fruit classes. For the *sq-rel*, Our network performs better in the banana class and slightly performs better in the other fruit classes. Similarly, Fig 5.25 compares the *RMSE* and $RMSE_{log}$ of each class of the fruit dataset. The proposed network performs better on the banana, orange and lemon class using the *RMSE* metric and compares with (Kuznietsov et al., 2017) on the apple and avocado class. For the $RMSE_{log}$, this network outperforms in the apple, avocado, banana and lemon classes.

5.7.2 Pose Estimation results

A total of 20 test frames are sampled for the 6DoF pose estimation and compared to the ground truth and predicted poses. The translation $[t_x, t_y, t_z]^T$ and the quaternion $[Q_x, Q_y, Q_z, Q_w]^T$ were compared for all fruit classes as shown from Fig 5.26 - 5.35. The ground truth poses are plotted alongside the prediction to visualise the deviation between the two. The ground truth is obtained as the relative pose provided between the camera and the fruits using the tracking system which is represented by red on the figures. The prediction on the other hand is the output from the TransPose model which is represented by green on the figures. It can be observed that the proposed network prediction compares well with the ground truth pose (translation and rotation) across all the fruit classes.

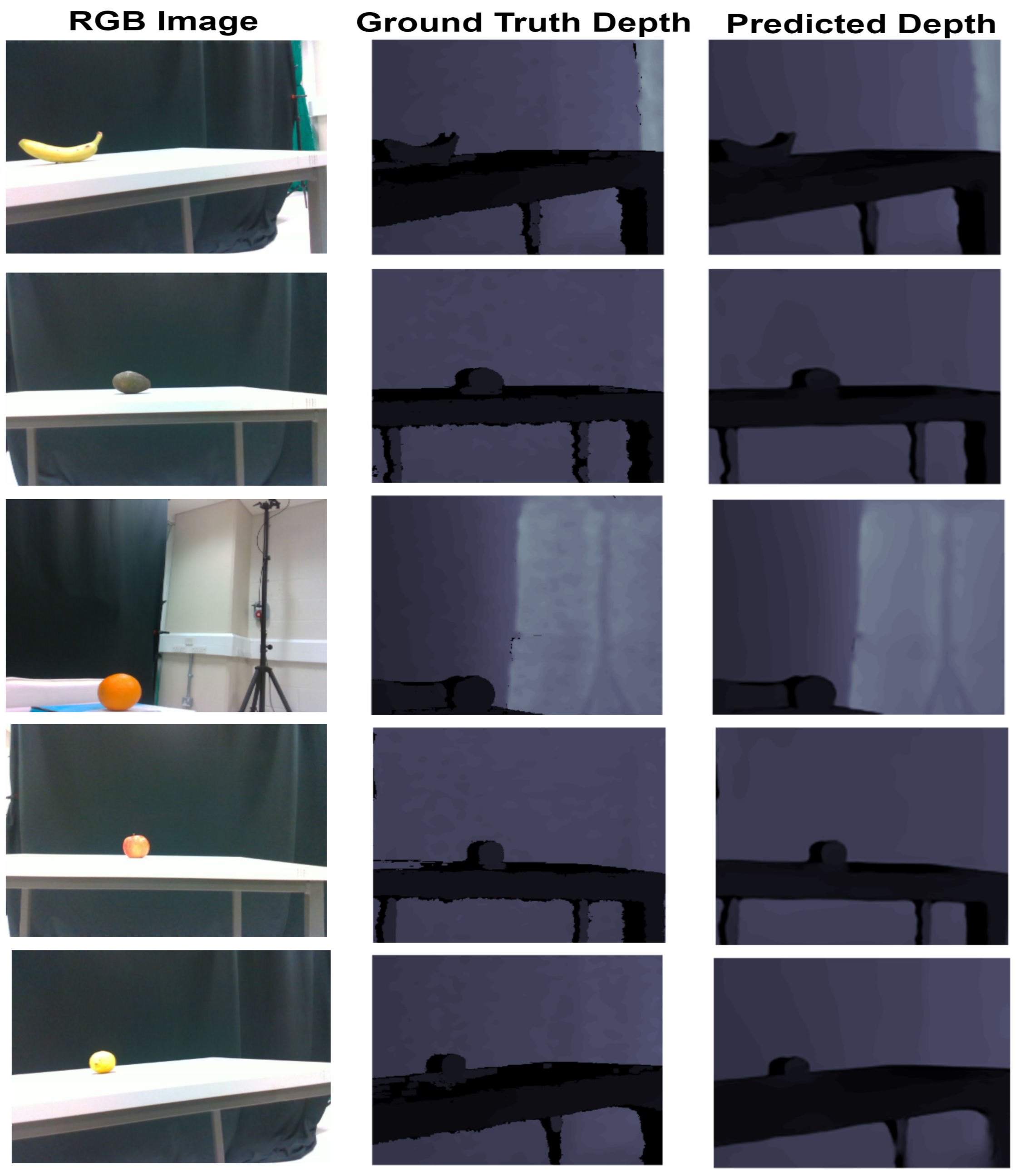


Fig. 5.23 Qualitative results of the depth prediction network. The left column shows the ground truth RGB Images of 5 classes. The middle column shows the ground truth depth images of the corresponding RGB images. The right column shows the predicted depth images from our network.

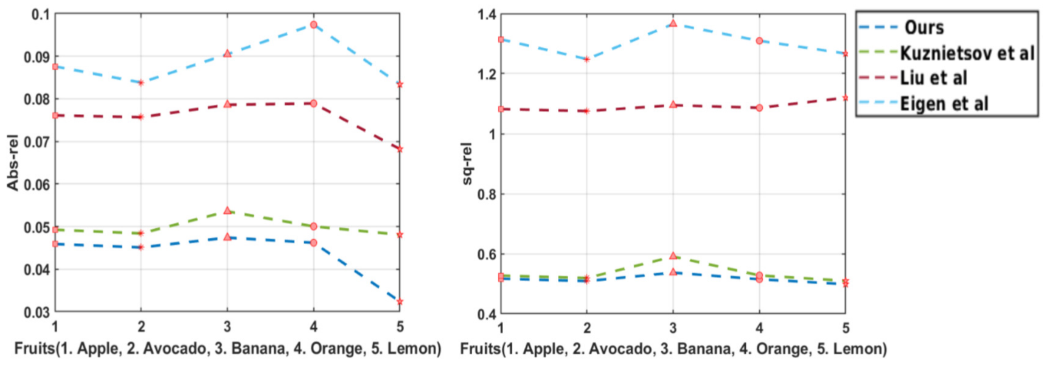


Fig. 5.24 *Abs-rel* and *sq-rel* metric comparison with other methods in the literature. Each fruit class with the corresponding evaluation metric is shown.

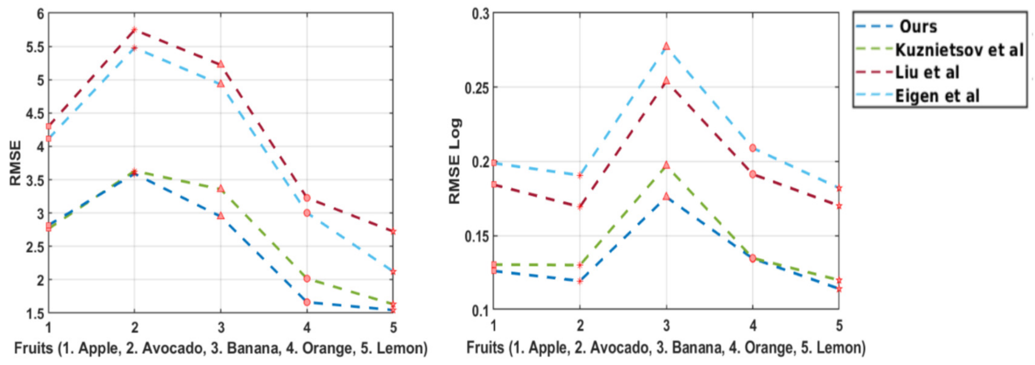


Fig. 5.25 *RMSE* and $RMSE_{log}$ metric comparison with other methods in literature. Each fruit class with the corresponding evaluation metric is shown.

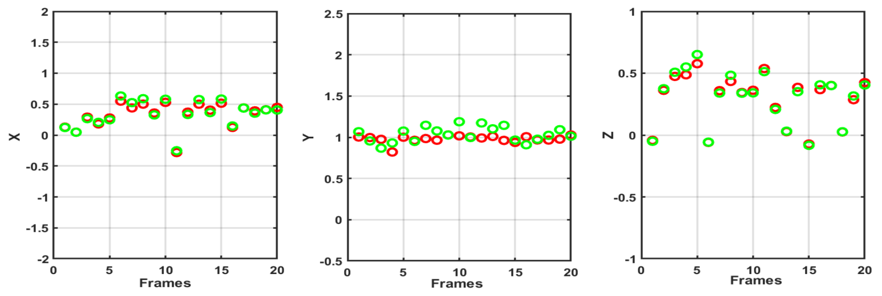


Fig. 5.26 Translation $[t_x, t_y, t_z]^T$ across 20 frames for apple fruit class. Red is the ground truth while green is the prediction.

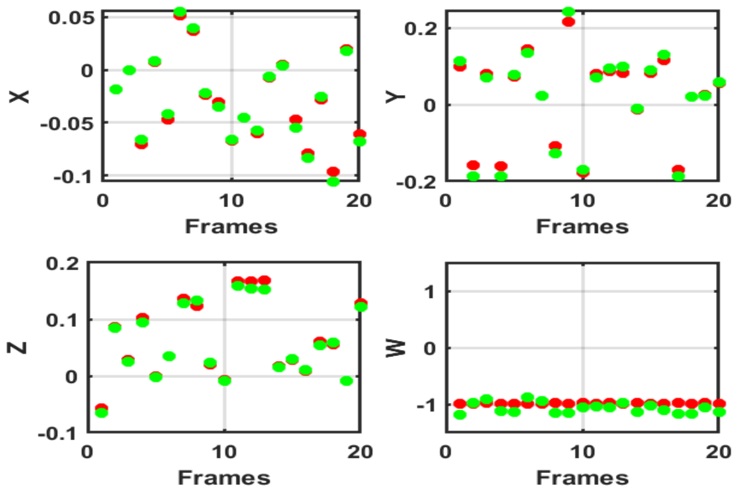


Fig. 5.27 quaternion $[Q_x, Q_y, Q_z, Q_w]^T$ across 20 frames for avocado... apple fruit class. Red is the ground truth while green is the prediction.

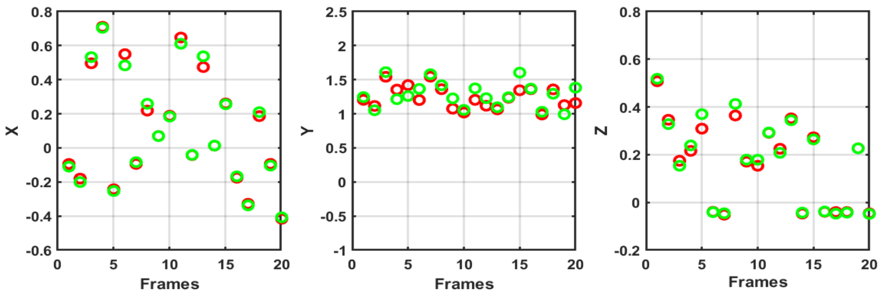


Fig. 5.28 Translation $[t_x, t_y, t_z]^T$ across 20 frames for avocado fruit class. Red is the ground truth while green is the prediction.

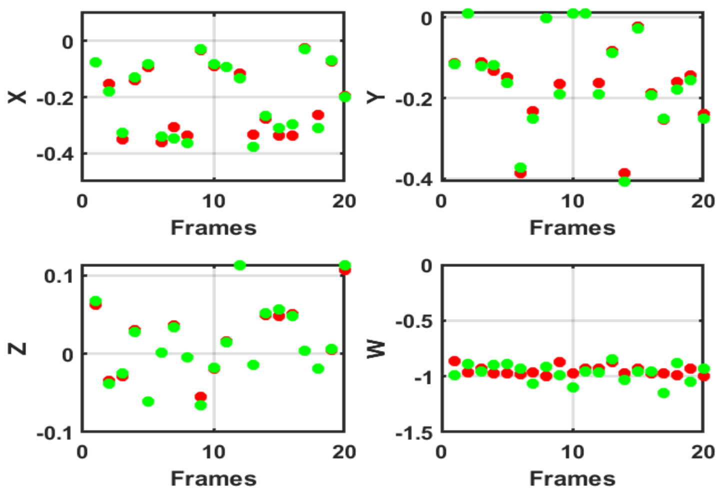


Fig. 5.29 quaternion $[Q_x, Q_y, Q_z, Q_w]^T$ across 20 frames for avocado fruit class. Red is the ground truth while green is the prediction.

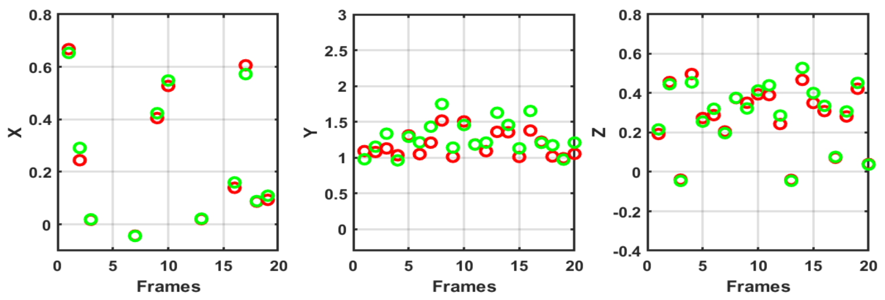


Fig. 5.30 Translation $[t_x, t_y, t_z]^T$ across 20 frames for banana fruit class. Red is the ground truth while green is the prediction.

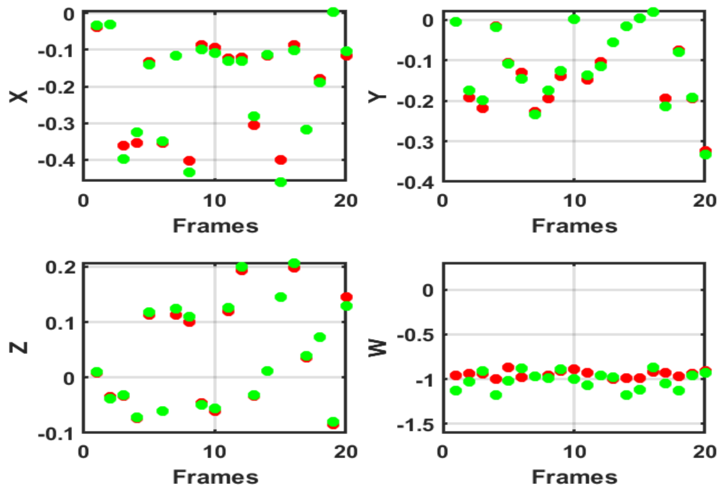


Fig. 5.31 quaternion $[Q_x, Q_y, Q_z, Q_w]^T$ across 20 frames for banana fruit class. Red is the ground truth while green is the prediction.

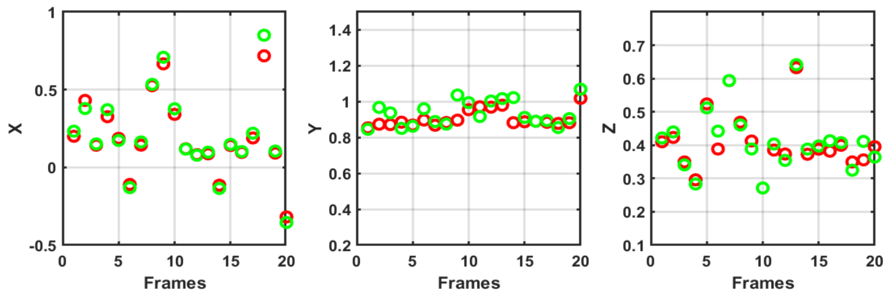


Fig. 5.32 Translation $[t_x, t_y, t_z]^T$ across 20 frames for lemon fruit class. Red is the ground truth while green is the prediction.

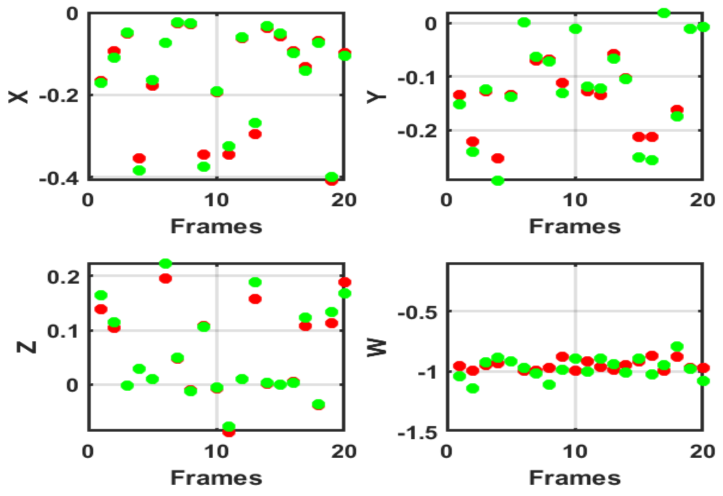


Fig. 5.33 quaternion $[Q_x, Q_y, Q_z, Q_w]^T$ across 20 frames for lemon fruit class. Red is the ground truth while green is the prediction.

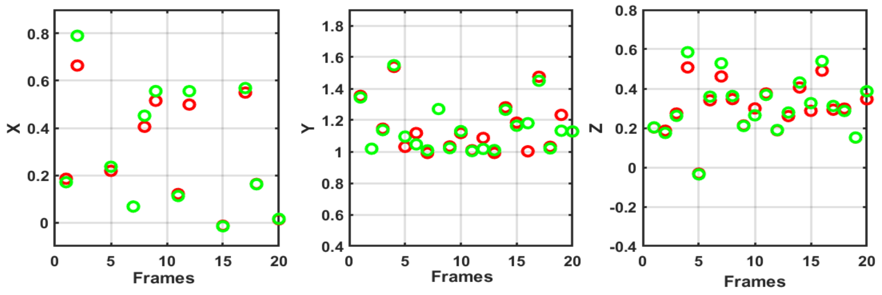


Fig. 5.34 Translation $[t_x, t_y, t_z]^T$ across 20 frames for orange fruit class. Red is the ground truth while green is the prediction.

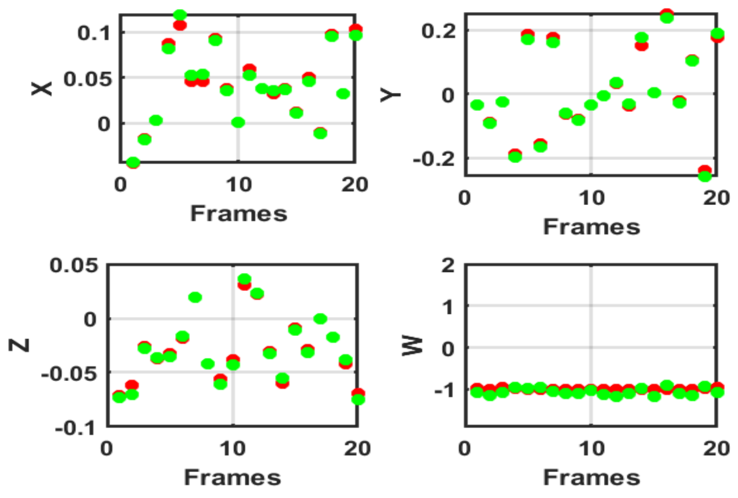


Fig. 5.35 quaternion $[Q_x, Q_y, Q_z, Q_w]^T$ across 20 frames for orange fruit class. Red is the ground truth while green is the prediction.

The qualitative result from some sample frames from the fruit dataset is shown in Fig 5.36. It shows different frames of different fruit classes with their respective bounding boxes.

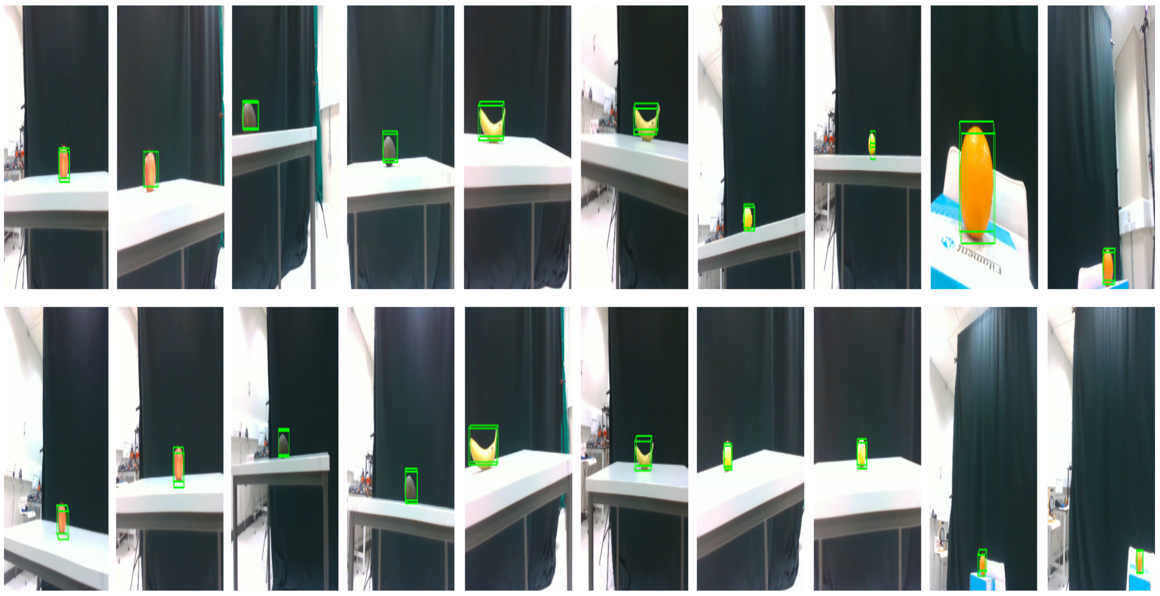


Fig. 5.36 Qualitative samples from the fruit dataset across all the classes

Table 5.4 shows a detailed evaluation of some objects from the YCB dataset using the metric in Eq. 5.20 and Eq. 5.21.

Table 5.4 Pose estimation Comparison using various approaches on some objects from YCB-V Dataset. Symmetrical objects are highlighted in red

ADD (Higher is better)			
Object	T6D-Direct	PoseCNN	TransPose
mug	72.1	57.7	75.7
tuna fish can	59.0	70.4	60.2
sugar box	81.8	68.6	84.5
bowl	91.6	69.7	89.7
master chef can	61.5	50.9	63.4
tomato soup can	72.0	66.0	75.6
wood block	90.7	65.8	90.7
pudding box	72.7	62.9	78.3
banana	87.4	91.3	90.4
bleach cleanser	65.0	50.5	70.2
ADD-S (Higher is better)			
mug	89.8	78.0	90.1
tuna fish can	92.2	87.9	91.7
sugar box	90.3	84.3	93.1
bowl	91.6	69.7	92.3
master chef can	91.9	84.0	92.4
tomato soup can	88.9	80.9	90.8
wood block	90.7	65.8	90.6
pudding box	85.1	79.0	88.1
banana	93.8	85.9	94.5
bleach cleanser	83.0	71.9	84.3
Mean (ADD)	75.38	65.38	77.87
Mean (ADD-S)	88.50	80.44	91.52

It can be observed that the TransPose model outperforms the other methods using the ADD metric in all the objects except the *tuna fish can*, *bowl*, *wood block* and *banana* where the network closely compares with the other methods. Similarly, using the ADD-S metric, TransPose outperforms the other methods except for the objects *tuna fish can* and *wood block*.

A similar comparison was conducted for the fruit dataset using the ADD and ADD-S metrics and the results are summarised in Table 5.5.

Table 5.5 Pose estimation Comparison using various approaches on some objects from YCB-V Dataset. Symmetrical objects are highlighted in red

ADD (Higher is better)			
Object	T6D-Direct	PoseCNN	TransPose
Apple	78.7	62.4	82.4
Avocado	81.3	71.4	82.6
Banana	90.4	76.6	92.4
Orange	71.4	59.7	79.3
Lemon	89.5	71.9	89.8
ADD-S (Higher is better)			
Apple	87.5	73.2	89.7
Avocado	86.2	82.9	92.6
Banana	92.7	82.3	93.2
Orange	84.6	80.2	87.8
Lemon	91.5	83.6	94.3
Mean (ADD)	82.26	68.40	85.30
Mean (ADD-S)	89.73	78.74	90.79

The mean from Table 5.4 and Table 5.5 shows the overall performance of the TransPose model across the sample objects. From the mean ADD and ADD-S, We can deduce that the depth refinement module improves the performance of 6 DoF pose estimation.

5.8 Conclusion and Future Work

This chapter proposed TransPose, an improved transformer-based 6DoF pose estimation network that utilises a depth refinement module to improve the overall performance. In contrast to other multi-modal networks that require two inputs, the TransPose network utilises an RGB image for the 6D pose estimation and the depth refinement with the aid of a depth estimation network. The 6D poses are directly regressed using the transformer network and further refined with the depth network. The results of the depth network are compared with other methods using the standard evaluation metrics. The performance of the depth network satisfies the purpose of 6D pose refinement. The results obtained using the standard evaluation metrics show a competitive outcome. Furthermore, A multi-modal fruit dataset was specially curated purposely for 6D pose estimation which is the first of its kind. The dataset encompasses an RGB image modality, depth image modality and 6D pose ground truth obtained from a tracking system. Thus, TransPose results are also evaluated on the fruit dataset to facilitate fruit-picking applications. TransPose solves the problem of pose estimation even when the platform is not aerial as in Chapter 4. It furthermore addresses the concerns of agricultural activities that require not only translation data but also rotation. This work shows the possibilities of AI in simplifying the pose estimation problem that finds applications in most agriculture activities (see Section 1.4) using a cheap monocular camera as a tool. In the future, the dataset can be enriched with more fruit classes and also provide more scenarios such as outdoor, leaves-occluded frames and trajectories. Also, the real-time onboard deployment of TransPose in conjunction with a manipulator for real-time fruit picking application can be explored by integrating the TransPose model into the pipeline.

Chapter 6

A Virtual Reality Teleoperation Pipeline for Precision Agriculture

Chapter abstract

Following the success of the TransPose model presented in Chapter 5, the idea of robotics teleoperation from a virtual environment becomes possible. Virtual reality as a tool provides an immersive experience in a 3D environment that allows users to visualise and interact with robots in such environments. Users benefit from better situational awareness to effectively perform remote tasks. In this chapter, we discuss a seamless teleoperation pipeline that interfaces virtual reality with robots for precision agriculture tasks. This pipeline allows users to visualise and control robots from a virtual environment thereby giving a telepresence advantage to users. ROS is utilised to accept data from a Unity gaming platform. The TransPose model is an essential component of the pipeline and is incorporated into the framework to estimate the 6DoF pose of the target fruit in real-time. We conducted an assessment of the 6DoF pose application that estimates and manipulates the pose of fruit targets within a virtual world. The goal is to mimic real-world actions to successfully grasp or harvest the fruit.

6.1 Motivation

Industries are rapidly adopting robotic platforms to automate their processes. This is likened to the ability of such platforms to perform tasks with high precision with a lesser completion time. Tasks that are hazardous will require minimal physical involvement to minimise the risk of injuries. However, some of these tasks will require a substantial level of interaction especially if the environment is uncertain. Tasks such as military operations (Kot and Novák, 2018; Solanes et al., 2022), medical application (Guzmán Ortiz et al., 2021; Law et al., 2021; Lim et al., 2021), disaster management (Habibian et al., 2021; Saputra et al., 2021; Sun et al., 2021), space operation (Jia et al., 2022; Schuster et al., 2020; Yin et al., 2022) and complex agricultural operations will not benefit greatly from a fully autonomous system due to many uncertainties associated with the nature of the challenges. Tele-operated robots with humans in the loop is a more suitable approach to such tasks as humans are often able to assess the uncertainties in the environment and take the necessary actions. Agriculture is a key sector to human survival. However, the continuous labour shortage has resulted in the advent of robotic platforms to perform agricultural activities to ensure continuous food supply. The activities such as irrigation (Aronson, 2013), harvesting (Wei et al., 2014), seeding (Amrita et al., 2015), fertilizer application (Adamides et al., 2017), crop monitoring (Ishibashi et al., 2013) and crop handling (Han et al., 2016) are common tasks that have benefited from robotic approach. However, some tasks such as weeding and occluded crop harvesting can be complex with many uncertainties in nature and will require more human intervention and monitoring. Fig 6.1 shows the robotics platforms in both the real and virtual worlds.

Teleoperation involves the controlling of a system remotely from a distant location over some form of communication network. This increases human safety and provides the ability to explore potentially dangerous or unknown locations. Conventionally, the robot perception is obtained by streaming from a camera or creating a point cloud map to understand the robot scene. However, this does not allow the user some privileges like having the sensation of being in the scene to understand it properly. Telepresence gives a user the immersive experience of actually being in the scene of action. In this chapter, we present a seamless

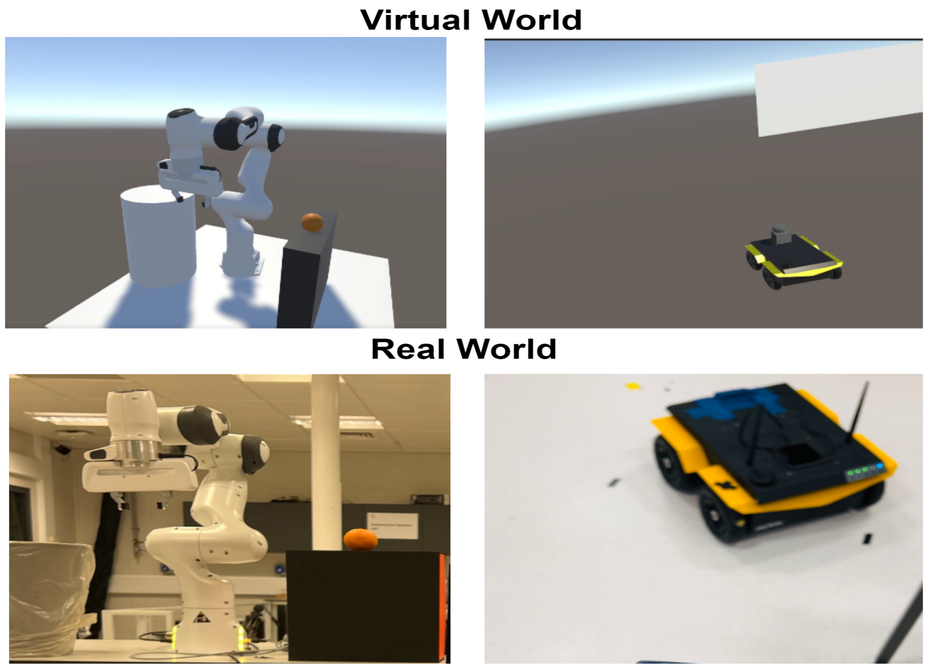


Fig. 6.1 Figure showing the set-up of the platform (Franka robotic manipulator and Jackal ground robot) in both Virtual and Real worlds.

teleoperation pipeline for virtual reality - robot interface applied to precision agriculture which we call *VitRob*. The VitRob pipeline has the following contributions:

- It is a seamless virtual reality - robot teleoperation pipeline that leverages ROS and allows users to control robots from a remote location.
- The pipeline allows users the benefit of telepresence in the form of a virtual world. This gives an immersive experience to users for better scene understanding and visualisation.
- The pipeline allows flexibility for integration with other functionalities. We integrated a 6DoF pose estimation nmodel for target detection and rendering.

6.2 Related Work

Automation of many tasks is nowadays easily implemented using Artificial Intelligence (AI). However, some applications are too complex and uncertain to fully automate and will often require humans in the loop. Robotics teleoperation is continuously evolving as researchers dwell into more adventurous and hazardous applications of robots. Research such as space exploration (Chen et al., 2019), underwater world (Brantner and Khatib, 2021; Sivčev et al., 2018), radioactive and aerial locations (Abi-Farraj et al., 2019; Bandala et al., 2019; Isop et al., 2019; Suarez et al., 2020) can benefit from this human-robot interactions. Some researchers have explored more intricate applications such as rescue missions (Kono et al., 2019) and medical surgeries (Chen et al., 2020; Saracino et al., 2020; Yoon et al., 2018). Many other contributions have been made to the implementation of human-robot teleoperation (Girbés-Juan et al., 2022; Lu et al., 2017; Nicolis et al., 2018; Selvaggio et al., 2018).

A new trend in teleoperation-based research is telepresence (Niemeyer et al., 2016). This gives the user a feeling of being in the scene of action. This concept is facilitated by virtual and augmented reality (Gorjup et al., 2019), haptic devices (Selvaggio et al., 2019), visual interfaces(Yoon et al., 2018) or a mixture of them (Clark et al., 2019; Girbes-Juan et al., 2020; Saracino et al., 2020).

In agriculture, the frequent change of environmental factors such as climate, farm terrain, crop growth stages and morphology can cause many uncertainties while modelling an automated system. To effectively perform some tasks under such conditions, human intervention is required at some point. Some works have been conducted to facilitate agricultural tasks using teleoperation technique (Adamides et al., 2013, 2014, 2017; Bechar and Edan, 2003; Godoy et al., 2010). This technique allows users to supervise the performance of complex tasks in uncertain conditions from a comfortable remote area.

The mentioned methods for teleoperation in agriculture do not necessarily give the user an immersive experience of the scene through virtual reality. As such, real-time understanding of the scene is not fully perceived resulting in uncertainties and low efficiency in task performance. Additionally, these methods are implemented strictly for a particular task and do not easily generalise to other applications.

6.3 VitRob Pipeline

The VitRob pipeline as seen in Fig 6.2 can be grouped into the following components:

- Virtual Environment
- Simulation Environment
- Real-time Environment
- TransPose
- ROS architecture

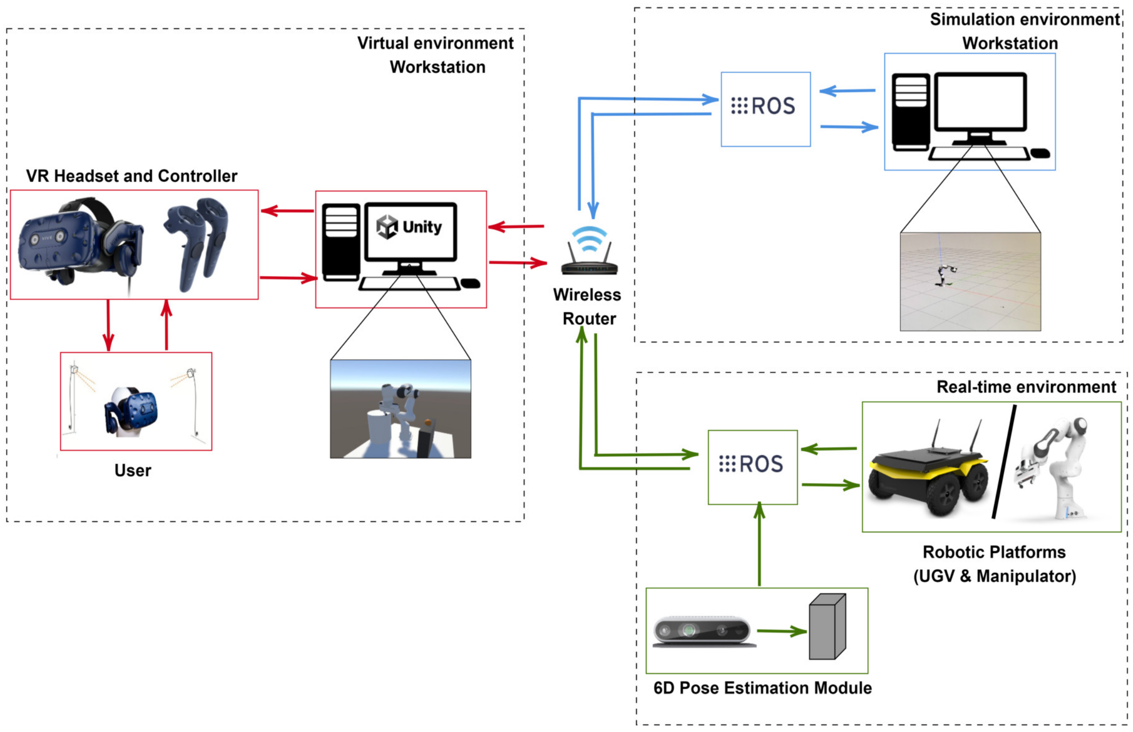


Fig. 6.2 Block Diagram showing the components of the VitRob pipeline and the connectivity between the various worlds (virtual, simulation and real-world)

6.3.1 Virtual Environment

The Virtual environment workstation encompasses the VR headset, controllers and a computer system. The user visualises the environment with the aid of an HTC Vive Pro headset (Vive, nd). It has a combined resolution of 4896×2448 pixels, 120° FOV and up to $120Hz$ refresh frequency rate.

The Virtual environment is developed using the Unity Platform (Technologies, nd). This allows us to model the robot and the accompanying sensors in the virtual world. Fig. 6.3 and 6.4 show the model of the robotic manipulator and the ground robot respectively. The Manipulator environment encompasses the robotic manipulator model with an accompanying end-effector and the target. The ground robot environment consists of the Jackal robot with the accompanying sensor (camera) and also the camera stream output for visualising the real-time robot camera feed. It is pertinent that the positions and dimensions of the modelled objects should correspond to the real objects as this gives a better scene understanding. These parameters such as robot pose, odometry, joints etc. can easily be updated through ROS with the aid of the ROS-sharp library (Martin, 2018) by transmitting them over a wireless network. The rendering of the objects then takes such parameters into consideration to render an accurate scene.

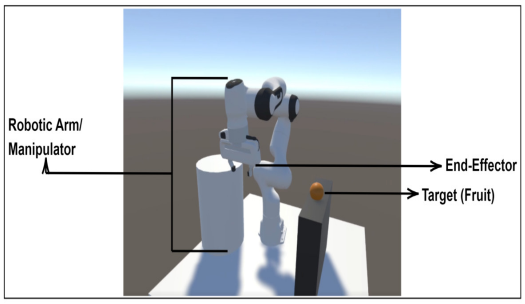


Fig. 6.3 Virtual Reality Set-Up for the Robotic Manipulator

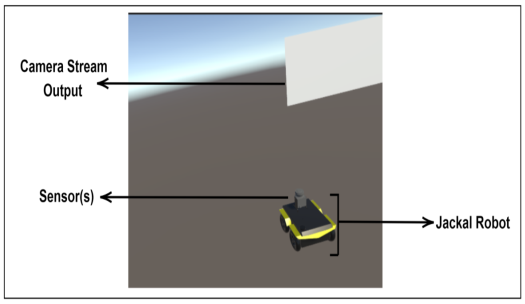


Fig. 6.4 Virtual Reality Set-Up for the Ground Robot

Unlike other approaches that use a third-party controller, our pipeline uses the generic VR headset controller. This way, the package will not depend on a third-party controller implementation but rather a plug-and-play solution. The keys on the controller are mapped to perform specific functions and thus the signals from the keys are transmitted wirelessly to the ROS server which is then further remapped to implement the desired action.

VR Hardware Setup

The setup consists of a headset, two controllers, a link box, and a base station. The headset shown in Fig. 6.5 is equipped with a Headset strap to hold the headset on the user's head, a tracking sensor to track the motion of the user, a camera lens to view the surroundings of the user, an Earphone for audio, Status light to indicate the status of the headset, Lens distance button to correct the lens distance, proximity sensor, and lenses to view the virtual world.

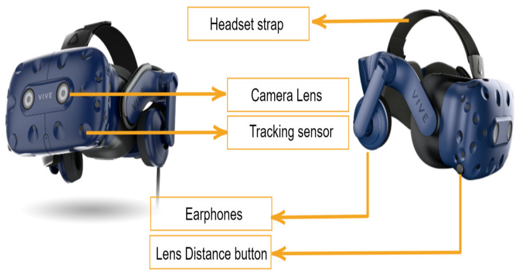


Fig. 6.5 VR headset and labeling

The VR controller shown in Fig. 6.6 has many buttons to accommodate multiple inputs from the user. It has a menu button, a trackpad, a start button, a grip button, a trigger, and motion sensors.

The VR link box in Fig. 6.7 is used to connect the headset to a computer. It is powered through the power port with a power cable. The display port and the USB port are used to convey audio and video from the computer to the VR headset.

The VR base station in Fig. 6.8 is used to tracking the motion of the headset and the controllers. The front panel beams signals to the headset and tracks the movement of the headset so that the same motion is replicated in the virtual world.

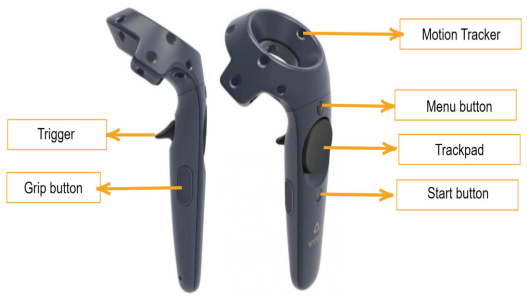


Fig. 6.6 VR controller and labeling

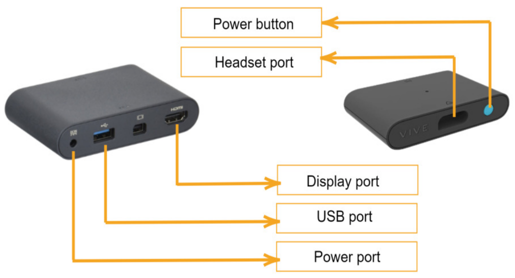


Fig. 6.7 VR link box and labeling

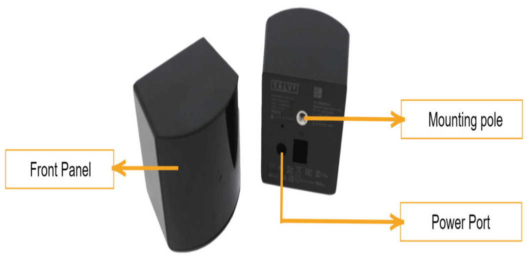


Fig. 6.8 VR base station and labeling

6.3.2 Simulation Environment

The pipeline provides a simulation avenue through the simulation environment. This allows users to test solutions before deploying them to real platforms and gives users a safety net to prevent accidents and equipment misuse. A computer with a Gazebo simulation environment (Gazebo, 2018) is utilised for this purpose. The model of both robots are shown in Fig. 6.9. The Jackal robot and Panda robot package (Christoph, 2020; Mike, 2020) are obtainable as open-source for development. The simulation equivalent of the intel real sense camera sensor is attached to test the image transmission pipeline. The important published topics such as the odometry, robot base, joint positions and image topic are transmitted to the unity platform over a WebSocket so that the virtual reality user can have access to them as seen in Fig. 6.12.

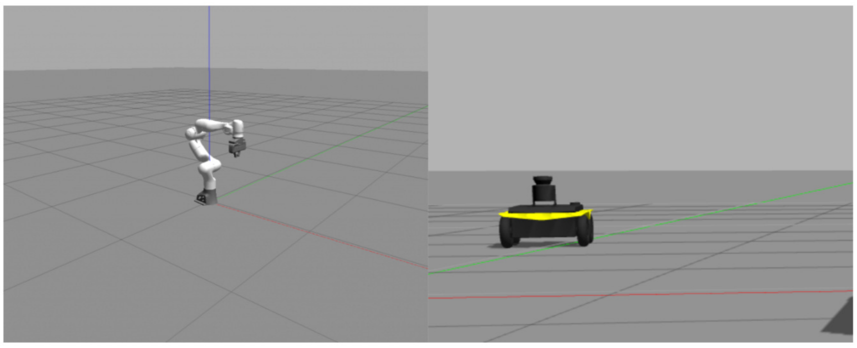


Fig. 6.9 Robotic Manipulator and Ground Robot Models in Gazebo Simulator

6.3.3 Real-time Environment

The physical robots (Jackal and Panda) are used for real-time teleoperation. The real-time setup for both platforms (Jackal and Panda) are shown in Fig. 6.10 and 6.11. The Jackal robot is a 4-wheeled portable and fast research ground vehicle. It has an Nvidia TX2 onboard processor, IMU and GPS integrated. It also provides support to additional sensors so users can easily utilise the system such as the intel real sense camera. The Panda manipulator is a 7 Degree Of Freedom robotic arm with a payload capacity of up to 3kg. All the robots served as a client on the wireless network and were interfaced with the virtual environment via unity by transmitting data through a WebSocket. The sensor data are published as rostopics on the host platforms. Users are then able to subscribe to the topics from the virtual environment workstation.

6.3.4 TransPose

TransPose is the transformer-based 6DoF pose estimation network earlier proposed in Chapter 5. The network is utilised in this pipeline to estimate the 6DoF pose of our target relative to the base of the robot. The target is spawned with the corresponding relative pose in the

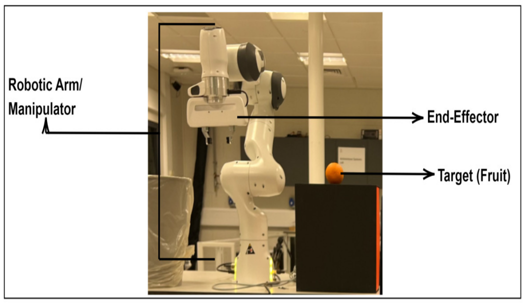


Fig. 6.10 Real-time Set-Up for the Robotic Manipulator

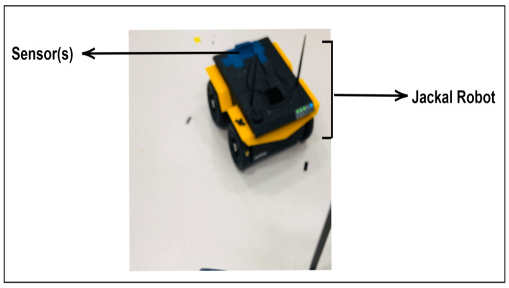


Fig. 6.11 Real-time Set-Up for the Ground Robot

virtual world so that the user can use the manipulator while visualising in the virtual world to acquire the target.

6.3.5 ROS architecture

The interfacing of the robots and the virtual reality is possible through ROS. The summary of the proposed architecture is shown in Fig. 6.12. The image topic is acquired from the camera and is utilised as the input for the 6DoF pose estimation via the TransPose model. The image and the estimated 6DoF pose are transmitted via the ROS bridge to the ROS sharp node running on the Unity platform. These topics are then subscribed and the images are rendered in real-time in the virtual world to allow the user to visualise the real scene of action. Due to limited bandwidth, we transmit the compressed versions of the images to allow other node instances to run smoothly and simultaneously.

The necessary data from the physical robots such as base, odometry, joints, and gripper is acquired by utilising their individual ROS packages. The data are then transmitted in the same manner to the virtual environment workstation. These data updates at a refresh rate of 40Hz and reflect on the robot spawned in the virtual world. These data are important for accurate scene construction and re-construction to allow users to have an up-to-date real-time situation.

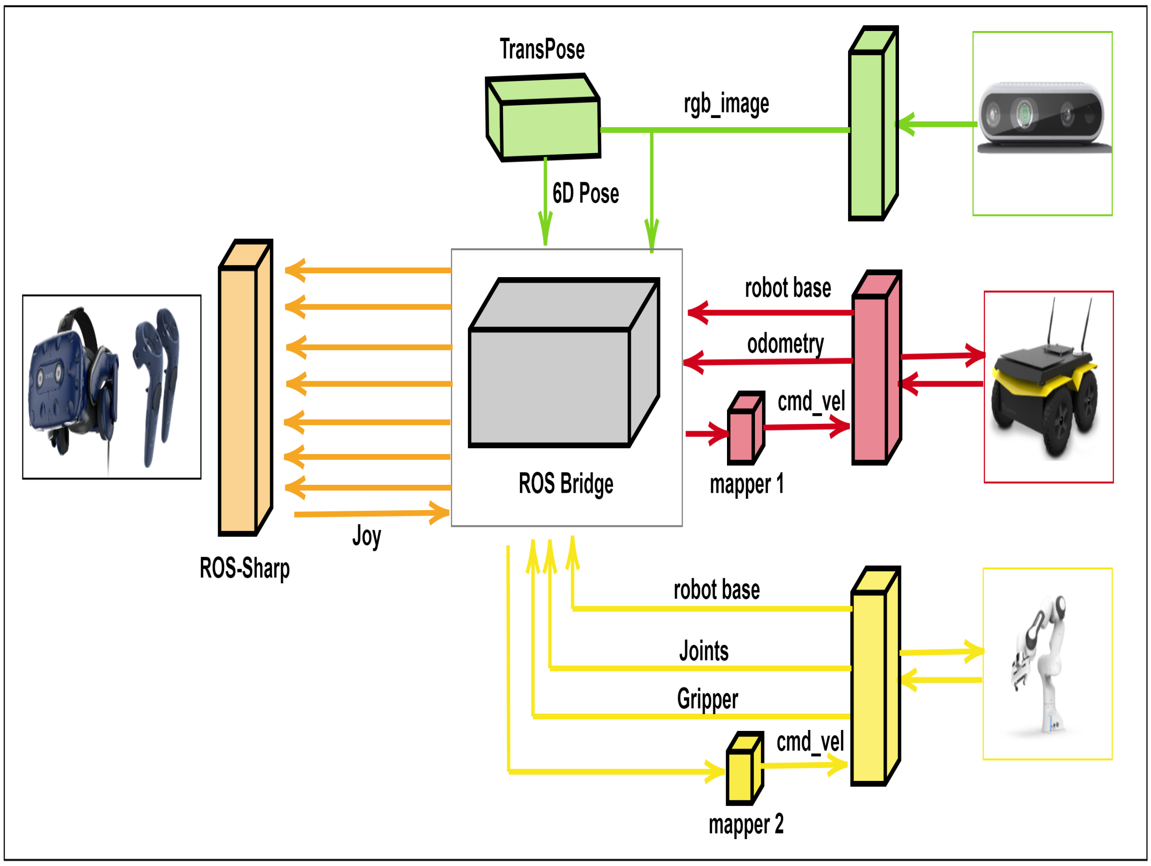


Fig. 6.12 Figure showing the ROS architecture including the messages published and subscribed to

6.4 Results

The results obtained are presented in three categories:

- Unmanned Ground Vehicle (UGV) Motion
- Target Position Estimation
- Robotic Manipulator Motion

6.4.1 Unmanned Ground Vehicle (UGV) Motion

The UGV is tracked in the real world while operating from the virtual world. The ground vehicle is navigated toward the target whose 6D pose is to be estimated. The experiment is repeated for 5 different scenarios and the motions were recorded as shown in Fig. 6.13 - 6.20. The scenarios are created by altering the initial position of the UGV and navigating it from within the virtual environment using the controllers towards the target positions.

Scenario 1 The UGV is placed at position $[X,Y] = [-0.08, 0.38]$. The UGV is then controlled from the virtual environment to a destination position $[X,Y] = [1.85, 0.65]$.

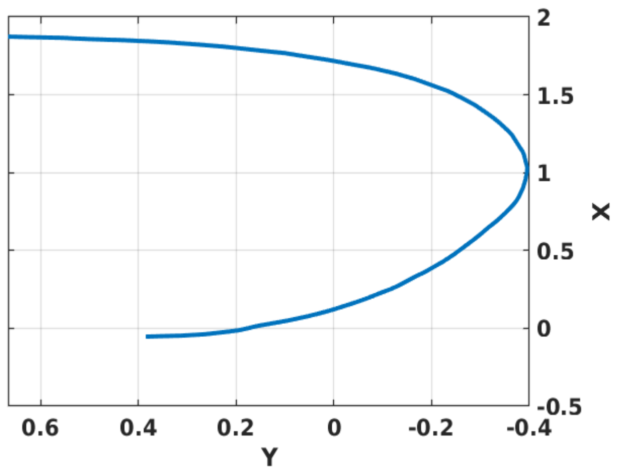


Fig. 6.13 Scenario 1, UGV translation [X, Y] in the real world while controlling from the virtual world to reach the target's position

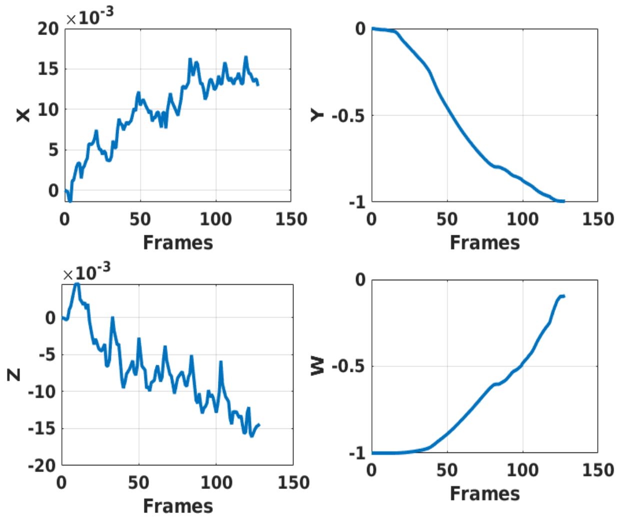


Fig. 6.14 Scenario 1, UGV quaternion [X, Y, Z, W] in the real world while controlling from the virtual world to reach the target's position

Scenario 2 The UGV is placed at position $[X,Y] = [1.48, -0.3]$. The UGV is then controlled from the virtual environment to a destination position $[X,Y] = [-0.08, 0.56]$.

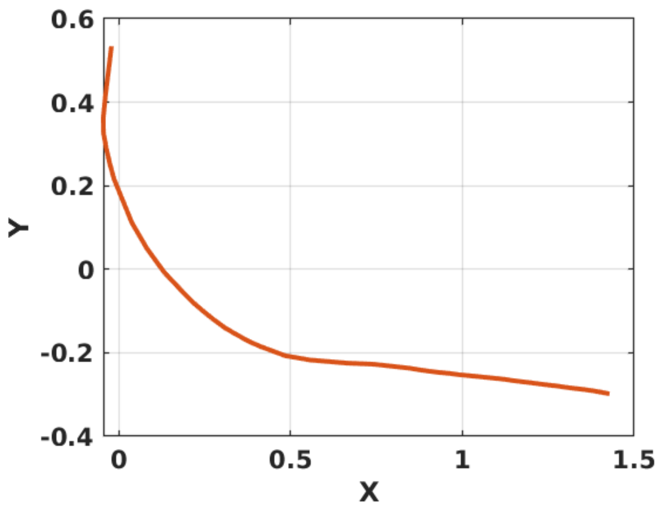


Fig. 6.15 Scenario 2, UGV translation [X, Y] in the real world while controlling from the virtual world to reach the target's position

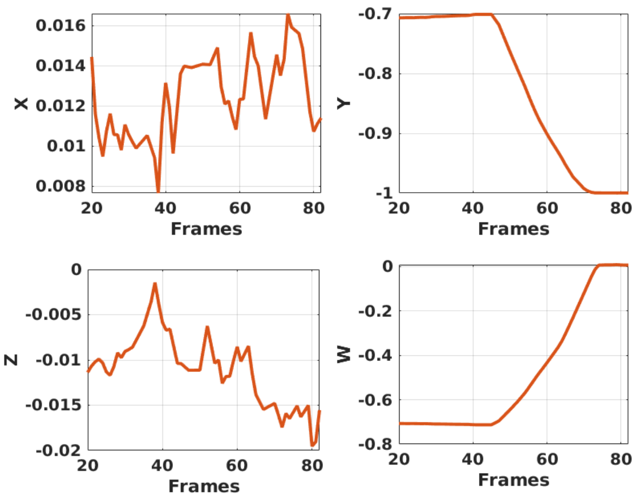


Fig. 6.16 Scenario 2, UGV quaternion [X, Y, Z, W] in the real world while controlling from the virtual world to reach the target's position

Scenario 3 The UGV is placed at position $[X,Y] = [0.8, -0.8]$. The UGV is then controlled from the virtual environment to a destination position $[X,Y] = [-0.9, 0.5]$.

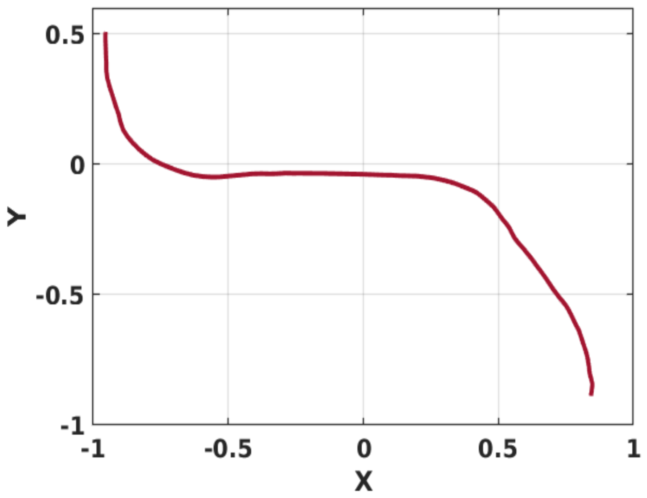


Fig. 6.17 Scenario 3, UGV translation [X, Y] in the real world while controlling from the virtual world to reach the target's position

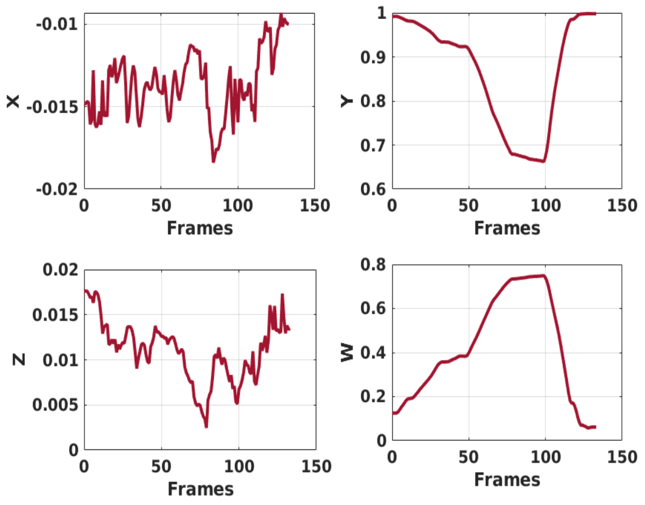


Fig. 6.18 Scenario 3, UGV quaternion [X, Y, Z, W] in the real world while controlling from the virtual world to reach the target's position

Scenario 4 The UGV is placed at position $[X,Y] = [1.05, 1.45]$. The UGV is then controlled from the virtual environment to a destination position $[X,Y] = [0.05, 0.5]$.

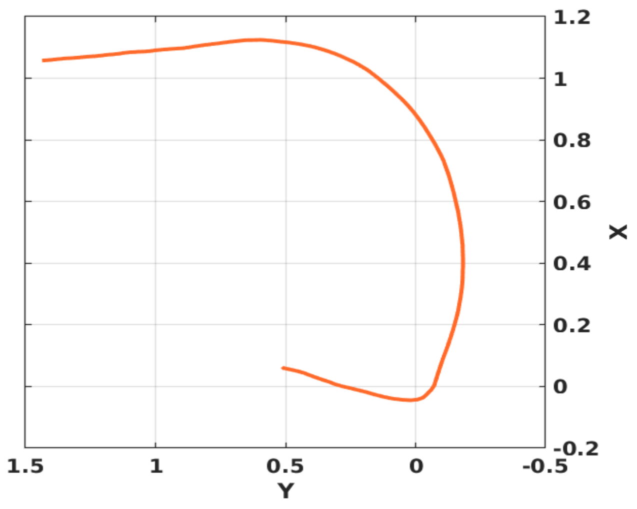


Fig. 6.19 Scenario 4, UGV translation [X, Y] in the real world while controlling from the virtual world to reach the target's position

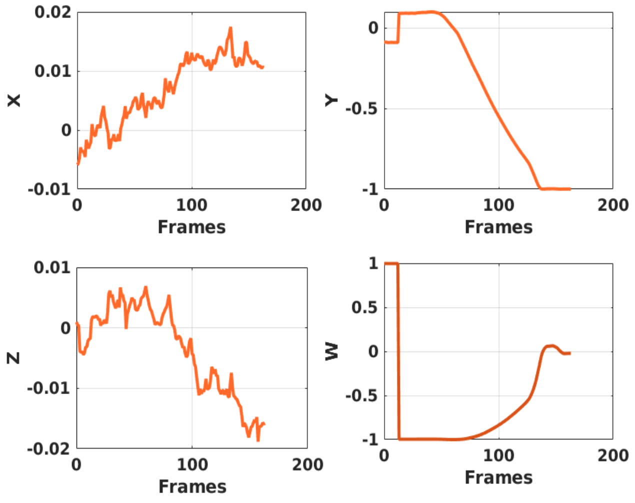


Fig. 6.20 Scenario 4, UGV quaternion [X, Y, Z, W] in the real world while controlling from the virtual world to reach the target's position

Scenario 5 The UGV is placed at position $[X,Y] = [-1.05, 0.35]$. The UGV is then controlled from the virtual environment to a destination position $[X,Y] = [0.65, -0.80]$.

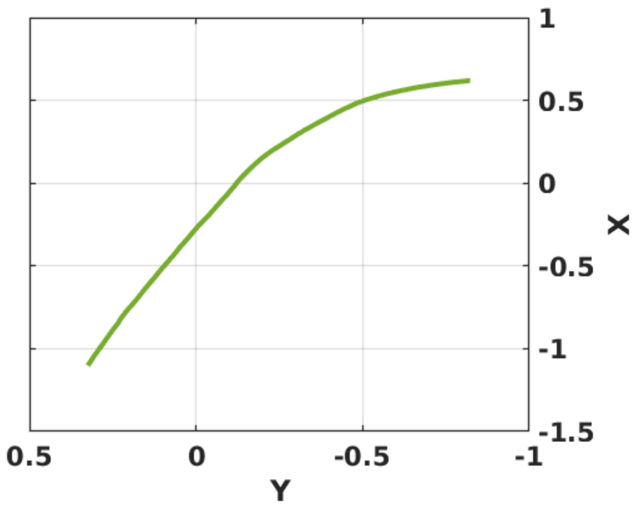


Fig. 6.21 Scenario 5, UGV translation [X, Y] in the real world while controlling from the virtual world to reach the target's position

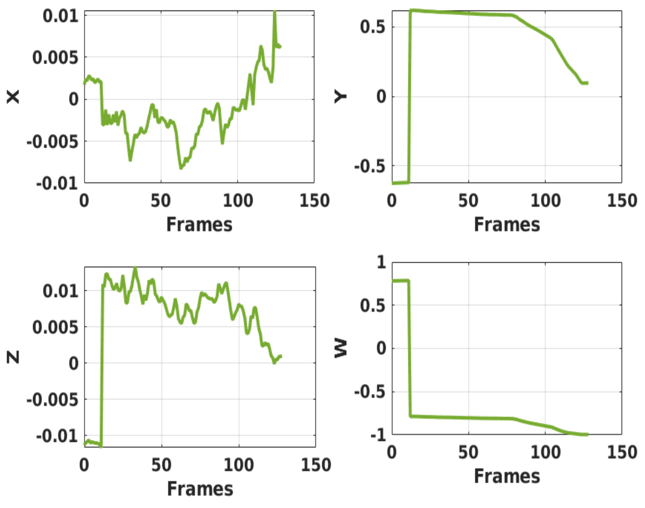


Fig. 6.22 Scenario 5, UGV quaternion [X, Y, Z, W] in the real world while controlling from the virtual world to reach the target's position

These results show the corresponding commands issued in the virtual world indeed are taking effect in the real world. The continued change of the position coordinates from the plots shows that the platform is not fixed but rather in continuous motion in consonant with the commands received. Since the ground truth of the destinations is known and mirrored in the virtual world, the user can simply use the controllers to control the UGV to the desired goal. Thus, we can deduce that the teleoperation of the UGV from within the virtual world has been successfully achieved.

6.4.2 Target Position Estimation

Upon reaching the target, the TranPose neural network pipeline is utilised to estimate the actual 6DoF pose of the target. The estimated pose is used to spawn the target in the virtual world. This is essential because the Unity virtual environment uses the 6DoF pose to initialise the target's position in the virtual world. The target's relative pose to the position of the robotic manipulator is then used to grasp the target in the virtual world. The belief is that the same action is being implemented in reality thus teleoperating the platform. The estimated results for the 5 scenarios are presented in table 6.1.

Table 6.1 Table showing the 6DoF pose (translation and quaternion) ground truth and estimation of the target

Scenario	Translation							
	Ground Truth				Estimated			
	X	Y	Z	-	X	Y	Z	-
1	-0.350	0.700	0.930	-	-0.361	0.720	0.912	-
2	-0.550	0.750	0.930	-	-0.581	0.780	0.93	-
3	0.500	0.920	0.930	-	0.493	0.924	0.931	-
4	0.450	1.200	0.930	-	0.450	1.180	0.940	-
5	0.400	0.520	0.930	-	0.400	0.518	0.913	-
Scenario	Quaternion							
	Ground Truth				Estimated			
	X	Y	Z	W	X	Y	Z	W
1	0.035	-0.320	-0.045	-0.945	0.034	-0.330	0.044	-0.947
2	0.044	-0.580	-0.035	-0.825	0.046	-0.591	-0.037	-0.830
3	0.050	0.620	-0.050	-0.750	0.051	-0.610	-0.048	-0.750
4	0.025	0.800	-0.010	-0.610	0.030	0.799	-0.010	-0.620
5	0.060	0.600	-0.030	-0.820	0.060	0.580	-0.030	-0.830

The errors in the estimations are shown in Fig. 6.23 - 6.29. The error is computed as the absolute difference between the ground truth pose and the estimated pose.

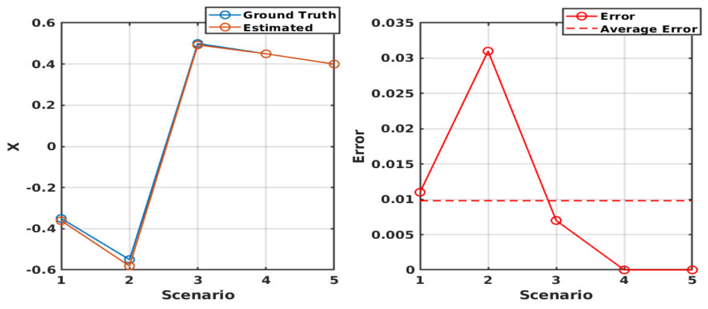


Fig. 6.23 Error in Translation along X-axis

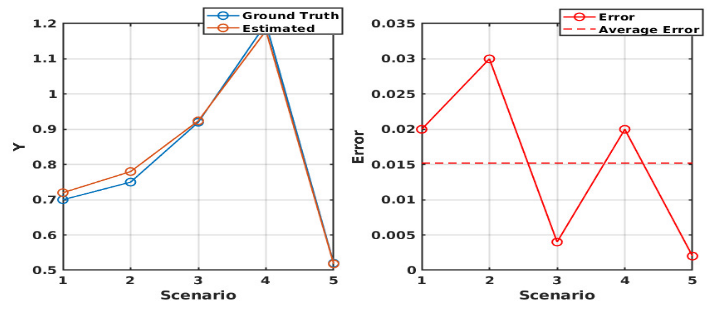


Fig. 6.24 Error in Translation along Y-axis

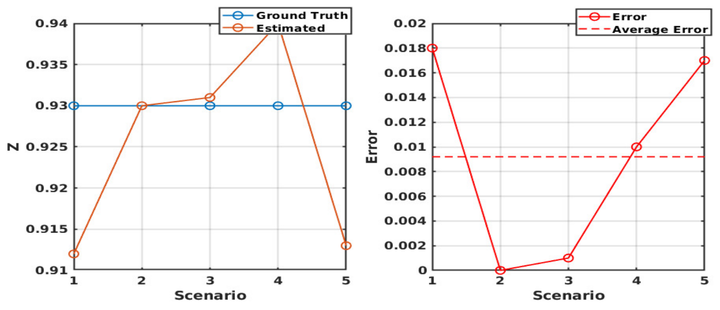


Fig. 6.25 Error in Translation along Z-axis

The average error is computed as the mean of all the errors obtained across all the scenarios. The average error obtained for the translation is 0.0098m, 0.0152m and 0.0092m along X,Y and Z axis respectively.

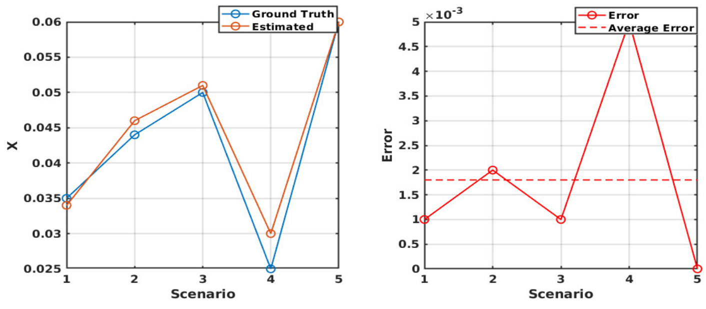


Fig. 6.26 Error in Quaternion Value X

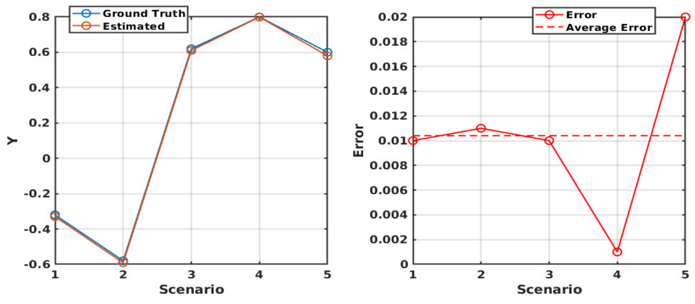


Fig. 6.27 Error in Quaternion Value Y

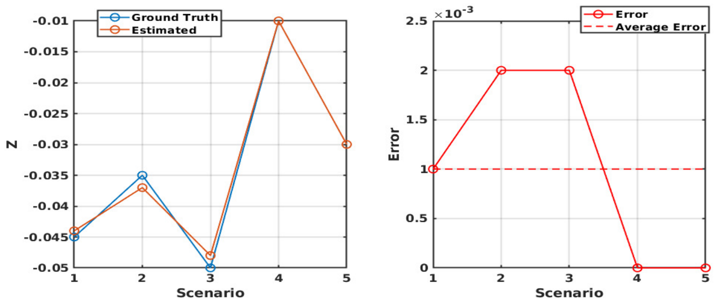


Fig. 6.28 Error in Quaternion Value Z

The average error obtained for the quaternion values X,Y,Z and W is 0.0018m, 0.0105m, 0.001m and 0.0058m respectively.

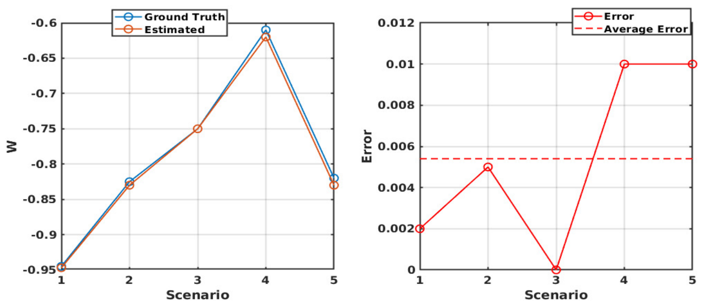


Fig. 6.29 Error in Quaternion Value W

6.4.3 Robotic Manipulator Motion

Finally, the trajectories are implemented by the robotic manipulator to grasp the target. These trajectories are observed and recorded so they can easily be evaluated. The trajectory is manually implemented in the virtual world since the target's 6DoF pose has already been estimated by the camera and the target's model has been spawned to the estimated location in the virtual world. Thus, while performing the grasping in the virtual world, the same manoeuvre is implemented in the real world. Fig. 6.30 - 6.39 show the trajectories for the 5 scenarios implemented for grasping the fruit.

Scenario 1 The end-effector is controlled from position $[X,Y,Z] = [0.1, 0.55, 1.0]$ from the virtual environment to a destination position $[X,Y,Z] = [-0.35, 0.70, 0.9]$. The plot in Fig. 6.30 shows the trajectory profile being implemented in the real time environment. The respective rotation profile throughout the motion are shown in Fig. 6.31.

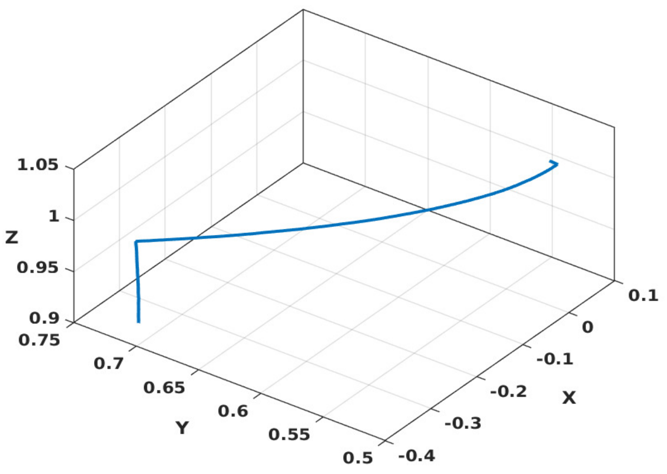


Fig. 6.30 Scenario 1, Manipulator's translation [X, Y, Z] in the real world while controlling from the virtual world to reach the target's position

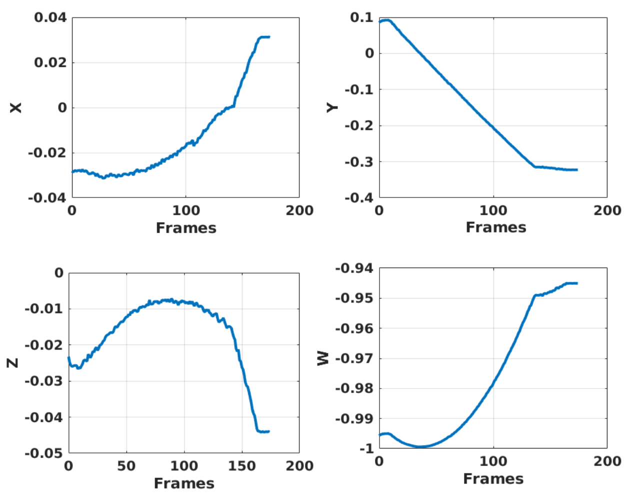


Fig. 6.31 Scenario 1, Manipulator's quaternion [X, Y, Z, W] in the real world while controlling from the virtual world to reach the target's position

Scenario 2 The end-effector is controlled from position $[X,Y,Z] = [-0.2, 0.8, 0.92]$ from the virtual environment to a destination position $[X,Y,Z] = [-0.59, 0.70, 0.92]$. The plot in Fig. 6.32 shows the trajectory profile being implemented in the real time environment. The respective rotation profile throughout the motion are shown in Fig. 6.33.

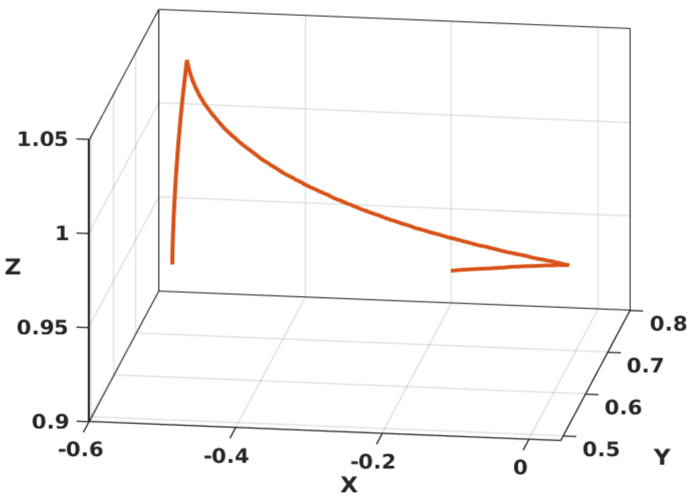


Fig. 6.32 Scenario 2, Manipulator's translation [X, Y, Z] in the real world while controlling from the virtual world to reach the target's position

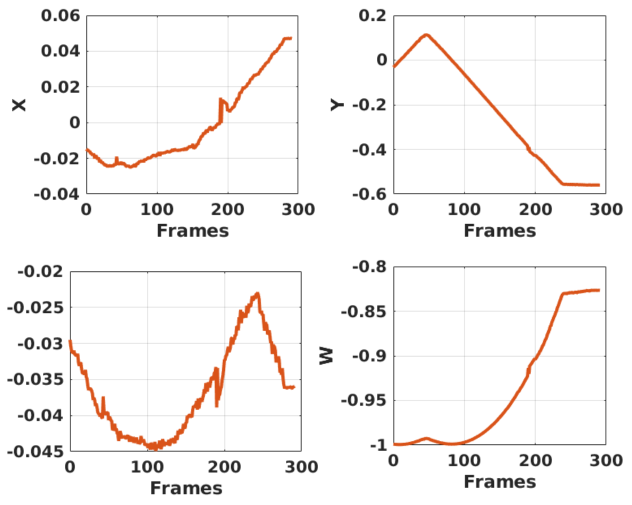


Fig. 6.33 Scenario 2, Manipulator's quaternion [X, Y, Z, W] in the real world while controlling from the virtual world to reach the target's position

Scenario 3 The end-effector is controlled from position $[X,Y,Z] = [0.08,0.8,0.94]$ from the virtual environment to a destination position $[X,Y,Z] = [0.45,0.87,0.94]$. The plot in Fig. 6.34 shows the trajectory profile being implemented in the real time environment. The respective rotation profile throughout the motion are shown in Fig. 6.35.

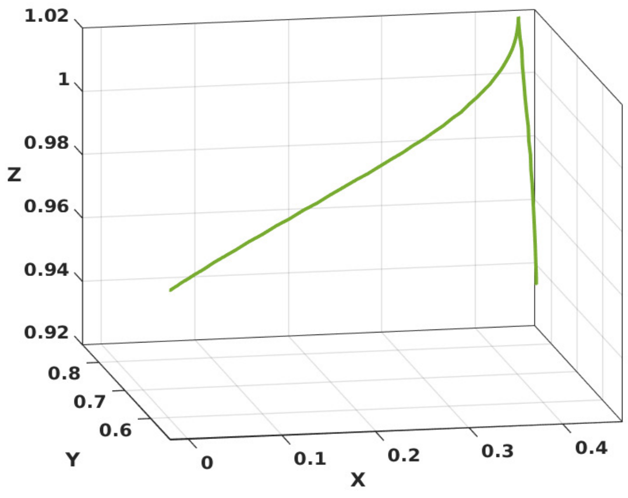


Fig. 6.34 Scenario 3, Manipulator's translation [X, Y, Z] in the real world while controlling from the virtual world to reach the target's position

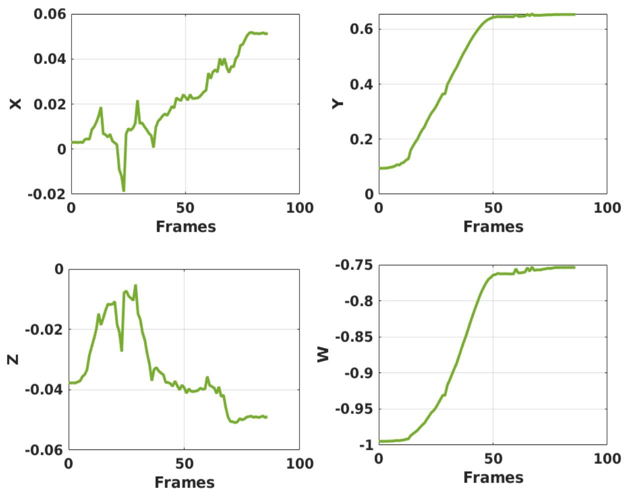


Fig. 6.35 Scenario 3, Manipulator's quaternion [X, Y, Z, W] in the real world while controlling from the virtual world to reach the target's position

Scenario 4 The end-effector is controlled from position $[X,Y,Z] = [0.2,0.5,0.95]$ from the virtual environment to a destination position $[X,Y,Z] = [0.45,1.19,0.95]$. The plot in Fig. 6.36 shows the trajectory profile being implemented in the real time environment. The respective rotation profile throughout the motion are shown in Fig. 6.37.

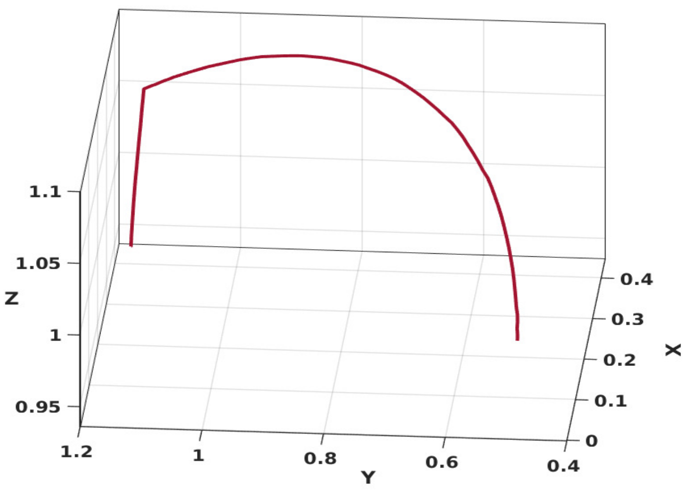


Fig. 6.36 Scenario 4, Manipulators translation [X, Y, Z] in the real world while controlling from the virtual world to reach the target's position

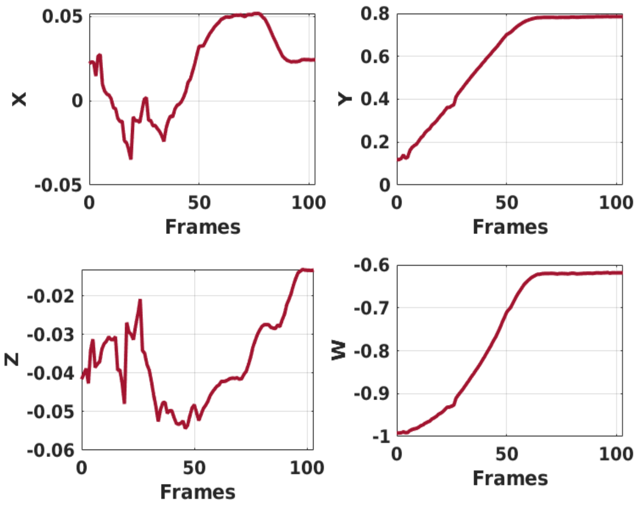


Fig. 6.37 Scenario 4, Manipulator's quaternion [X, Y, Z, W] in the real world while controlling from the virtual world to reach the target's position

Scenario 5 The end-effector is controlled from position $[X,Y,Z] = [0.0, 0.55, 1.0]$ from the virtual environment to a destination position $[X,Y,Z] = [0.45, 0.52, 0.92]$. The plot in Fig. 6.38 shows the trajectory profile being implemented in the real time environment. The respective rotation profile throughout the motion are shown in Fig. 6.39.

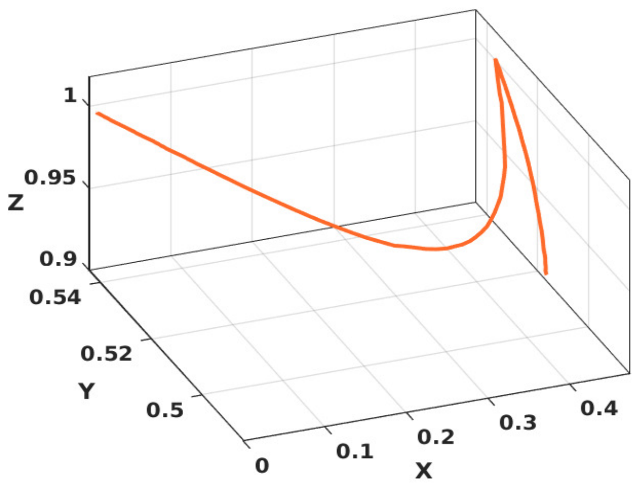


Fig. 6.38 Scenario 5, Manipulator's translation [X, Y, Z] in the real world while controlling from the virtual world to reach the target's position

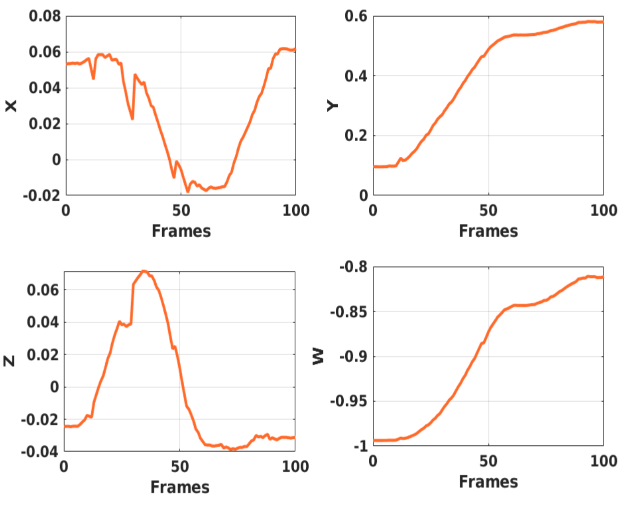


Fig. 6.39 Scenario 5, Manipulator's quaternion [X, Y, Z, W] in the real world while controlling from the virtual world to reach the target's position

These trajectories show that the robotic manipulator is in motion and moves continuously toward the target as intended by the user from the virtual environment. The tracking data shows how the user implements the trajectory in the virtual world and thus reflects in reality until the target is acquired.

6.5 Conclusion and Future Work

VitRob Pipeline, A seamless teleoperation pipeline for an advanced virtual reality-robot interface is proposed in this chapter. Although the pipeline is versatile, The chapter demonstrated the effectiveness of this pipeline by performing simple fruit harvesting/grasping motions in precision agriculture. The pipeline benefits users with an immersive experience in the form of a virtual world which allows the user to take a better-informed decision from a remote location. The transPose model earlier proposed in Chapter 5 is integrated to detect the fruit and estimate the position. The results of the motions performed in the virtual world and the Real world are observed and recorded. These motions executed from the virtual environment accurately match the real-world motion with minimal latency.

In the future, a DNN model can be investigated to perform a continuous real-time estimation while updating the virtual scene with little to no latency about sudden changes in the environment. This will provide users with a better telepresence experience in real time. Also, the collaboration between many platforms using this pipeline to simultaneously execute commands in a multi-agent scenario with no latency can be investigated.

Chapter 7

Conclusion

7.1 Overview

Agriculture has been a practice for humans since the existence of life. Humans always find a way to cultivate their food for livelihood and survival. The continuous increase in food demand as the world population grows is an important concern for the existence of human life. Especially in the modern world where the population are opting for white-collar jobs. Also, conventional crude tools are no longer efficient enough to meet the food demand. The labour intensity, time consumption and weakness associated with these crude tools can not be neglected amidst food scarcity.

As human evolves through time, so does their agricultural practices. In modern days, mechanised farming is considered a major breakthrough in agriculture. It involves the use of heavy machinery for agricultural practices such as planting, harvesting, tillage, weeding, etc. Although this method provides a substantial level of efficiency as compared to the use of crude tools, an argument surrounding the disadvantages of these machines can easily invalidate their potential. To begin with, the cost of acquiring such machines is very high and unaffordable to a common farmer. The negative environmental factors attached to these machines are also damaging to the ecosystem. Due to their weight, they can cause soil compaction and erosion. Most of the machines run on fossil fuels, releasing greenhouse gases into the atmosphere and depleting the environment. They also tend to destroy the immediate habitat of wildlife. Another factor to take into consideration is the technical know-how that is required to operate the machines. Since the machines are not equipped with a level of intelligence to perform tasks autonomously, a lot of effort is put into learning and operating these machines due to their complexity. Although agricultural activities are often distinct from each other, some points of confluence established in this thesis (see Section 1.4) as the common backbone or key elements of agricultural activities are *classification*, *detection* and *pose estimation*. The lack of these abilities is exhibited by the farm machines and thus renders them incapacitated for automation and generalisation.

Robotic platforms are often more sustainable, green, and efficient platforms for many applications ranging from medical, military, space and industrial activities. Robotics in agriculture offers the following advantages over mechanised farming:

- Efficiency: Robotic platforms can cover a large field in a short time while working round-the-clock.
- Precision: Due to their designs, they can perform tasks with high precision that can not be achieved using crude tools and heavy machinery.
- Sustainability: Robotics platforms do not run on fossil fuel and thus do not emit greenhouse gases. As they deal with precision, they reduce the wastage of harmful chemicals such as pesticides, herbicides, and insecticides on the environment.
- Data-Driven: Robotic platforms are equipped with multiple sensors that gather data. The data is used by the robot to implement tasks with high precision.
- Flexibility: A single robotics platform can be used for many agricultural tasks and can be applied to many varieties of crops.

Although the advantages of robotics platforms over heavy machines have been established, the notion of robots being able to tackle the three key elements (classify, detect and estimate pose) has not yet been established. Artificial intelligence allows systems to perform tasks that would have required human intelligence. They allow systems to reason and make decisions based on patterns and previous experience. Equipping such algorithms into robotic platforms will make a great system that will tackle the three key elements thus *automation*. These systems can be used autonomously to ease agricultural practices, enhance crop yield and sustain the environment at an affordable cost.

The thesis leveraged the potential of artificial intelligence AI and explored how such a tool can be used for precision agriculture. With AI for classification, detection, and Pose estimation as focus, The work successfully investigated these elements and proffered state-of-the-art methods to improve agricultural practices. Unmanned Aerial Vehicles (UAV), Unmanned Ground vehicles (UGV) and robotic manipulators equipped with the required affordable sensors were used to facilitate this research. From a conclusive point, the fusion between AI and robotics platform have been proven possible for precision agricultural practices in this thesis.

7.2 Summary and Discussion

The objectives of this thesis were established in Section 1.4 and further simplified into research questions (see 1.4), which are now reviewed as the work concludes with an overview of the novel contributions made by each Chapter and a discussion of potential future study.

The first contribution of this thesis is in the form of a deep network for weed classification and detection in precision agriculture in Chapter 3. This work explored both classification and detection by fusing a classification network (ResNet-50) and a detection model (YOLO) which we name fused-YOLO. From experimental results, weed types are able to be classified and assigned bounding boxes. The ability to isolate the object of interest (weed) with high precision will allow other build-up applications such as selective spraying/weeding. Selective spraying reduces the quantity of harmful chemicals that are released into the environment since a target within the FOV can be isolated and sprayed rather than a blanket spraying. Additionally, since not all types of chemicals work for every plant/weed, the proposed model comes into play when identifying (type) and isolating (detection) the specie of the intended target for selective spraying. From a sustainability point of view and as against mechanised farming, the work addresses the emission of greenhouse gases by utilising a

UAV which is clean and sustainable. Another important contribution is the fusion of this algorithm with an affordable drone to combat the issue of affordability. Moreover, the solution offers a good level of flexibility in the sense that it can easily be deployed to other platforms. Conclusively we can sufficiently imply that this work has addressed two research Questions (see Section 1.4) which reads: *Can a robot perform autonomous classification for agricultural applications in real-time? if yes how?.* and *Can a robot perform autonomous detection for agricultural applications in real-time? if yes how?.*

Chapter 4 extends the fused-YOLO model to relative position estimation of multiple weeds facilitated via UKF. The detection bounding boxes are utilised also to estimate the position of the detected targets. The solution is fast and uses an affordable UAV platform equipped with a monocular camera while providing a good estimation accuracy. To achieve this, firstly, the weed is classified and detected by a drone using the fused-YOLO pipeline. The detection bounding boxes are used to obtain the centre of the weed from the image frame. The centres are converted to bearing angles relative to the drone and are further used in a UKF estimation scheme to estimate the position of the detected target. This novel UAV-integrated deep network detection and relative position estimation scheme exhibited a fast converging time judging from the simulation and experimental results. Indeed, AI can extend beyond classification and detection. It can also combine with other mathematical tools like the UKF to perform other operations like target position estimation. Thus, a notable contribution of this thesis is also the combination of both tools in a robotic platform to facilitate precision agriculture through the pose estimation of detected weeds. A typical use case is in the event of a very large farm that will rather be inefficient for a UGV to patrol. This solution which is deployable on a UAV can simply fly over this large farm, detect the weeds, estimates their positions and send these positions to the UGV. The UGV simply navigates to that exact location as against patrolling the whole field thereby saving time and resources. Overall, the chapter attempts to address the research question that reads "*Can a robot perform autonomous pose estimation for agricultural applications in real-time? if yes how?*" and has done that successfully.

This thesis also contributes to the adoption of AI for 6DoF pose estimation in precision agriculture. A novel pipeline based on a transformer model is proposed and termed TransPose. As elaborated in Chapter 5, TransPose is an improved transformer-based 6DoF pose estimation network that utilises a depth refinement module to improve the overall performance of the pose estimation. Unlike most 6DoF pose estimation pipelines that leverage two multimodal inputs, TransPose leverages a single RGB image for the 6D pose estimation and henceforth extracts depth information from the same RGB image using a lightweight FPN. The depth refinement module utilises the estimated depth information to improve the overall accuracy of the 6DoF pose estimation. Competitive results were obtained using the standard evaluation metrics. Building upon the work in Chapter 4, challenges such as movability, platform dynamics and availability of space may emanate while adopting the proposed solution. Thus, the TransPose model addresses this concern by providing a solution that estimates 6DoF poses with a single image frame using AI, i.e. the need for performing a trajectory is completely eliminated. The application of this solution in precision agriculture is very versatile making the solution a robust one. Any application in precision agriculture involving relative contact between the platforms and the plants, crops etc (such as robots for harvesting, robots for weed removal, robots for planting, robots for pruning etc) can benefit from this solution. Again, this buttresses the assertion that indeed a robot can utilise AI for pose estimation there by answering the third research question.

As part of this research, A novel multi-modal fruit dataset called Fruity was acquired for 6DoF pose estimation. AI for precision agriculture is gradually maturing yet the availability of the dataset is still an issue. The Fruity dataset is another breakthrough contribution in the precision agriculture community as the dataset is the first ever to specifically target 6DoF pose estimation of fruits. The dataset is acquired with a rig of sensors mounted on multiple platforms (UAV and Robotic arm) and also handheld. Trajectories were performed to facilitate the recording of the outputs in 3 modalities. This consists of depth, RGB and thermal images. Additionally, the dataset provides the camera's and target's 6DoF poses for each frame of the data capture. The dataset is envisaged to have use cases such as fruit classification, fruit detection, fruit 6DoF pose estimation and fruit picking/harvesting application among others.

VitRob Pipeline, A seamless teleoperation pipeline for an advanced virtual reality-robot interface is another contribution of this thesis. This pipeline justifies the application of the earlier proposed TransPose for 6D pose estimation in fruit-picking applications for precision agriculture. The pipeline allows users to remotely harvest/pick fruit from a virtual environment with the aid of AI. The integrated TransPose pipeline detects and estimates the 6DoF pose of the target and renders it in the virtual environment. The user who is submerged in a virtual environment is then able to remotely teleoperate with a robotic manipulator to acquire the target. This novel pipeline for fruit picking/harvesting in precision agriculture also provides an immersive experience that can benefit users with better situational awareness due to telepresence capability through VR. The results of the motions performed in the virtual world and the Real world were observed and recorded with low latency.

Through well-researched methods and several experiments, this thesis has explored the possibilities of artificial intelligence (AI) on robotic platforms for precision agriculture. Although artificial intelligence integration with robotic platforms for precision agriculture can be very broad considering the numerous and versatile nature of agricultural activities, the thesis has comprehensively curated some crucial and interesting applications for artificial intelligence in precision agriculture. This is referred to in the thesis as the *key elements*. These elements form the backbone for the majority of precision agricultural activities as earlier established. Classification, detection, and pose estimation, are the elements forming the bedrock of the activities. Thus, this research has adequately addressed these key elements (and teleoperation) using artificial intelligence techniques and opened up more possibilities for researchers to explore the precision agriculture domain.

7.3 Future Work

This thesis proposed several methods of integrating AI in robotics platforms to facilitate precision agriculture. However, some research gaps are required to be covered in order to thoroughly enhance the exploitation of AI for precision agriculture. This section briefly covers some potential gaps that may require further investigation or propositions that can improve the methods presented in this thesis.

This thesis has utilised open sourced dataset as well as a novel custom-made dataset. The models proposed in this thesis all rely on a dataset for functionality. However, the quest for better accuracy can depend on the type and richness of the dataset used to train the models. For example, the deep detector/classifier model's accuracy proposed in Chapter 3 can be improved by training the model with a larger dataset. As seen also in section 5.4.4 that the richness of the dataset can influence the results, a well-distributed dataset with an even spread can help the accuracy of the trained models. Also, the detector/classifier models can be evaluated more with similar-looking species that have close characteristics to the used weeds.

For the custom-acquired dataset, the method of acquisition can impact the quality of the dataset. The dataset quality utilised can be enriched with more fruit classes with different geometrical appearances. Although the model in Chapter 5 was utilised for indoor scenarios, An evaluation can be conducted for outdoor scenarios which can be facilitated with outdoor-oriented datasets. The outdoor dataset can have uncontrolled illumination, varying backgrounds and leaves-occluded frames to make the evaluation even more interesting. The custom dataset was acquired on a robotic manipulator and a drone. Other platforms such as UGV can also be used to perform more intricate trajectories so as to cover more FOVs thereby enriching the dataset with more platforms and thus more applications.

Chapter 4 extended the use of the detector/classifier model for target position estimation by performing an elliptic trajectory. An elliptic trajectory was so chosen to cover a reasonable FOV during the estimation. However, optimizing the trajectory for the estimation can result in better FOV, and thus more targets can be estimated simultaneously. A study on the optimal trajectory for each scenario can also be conducted to facilitate faster and more accurate convergence. In the case of a monocular sensor, a light network that predicts depth can be incorporated into the pipeline. This will make the pipeline a more AI oriented technique and additionally, may provide competitive estimation results. Otherwise, an affordable stereo camera sensor can also be utilised to directly get the depth between the platform and the target rather than the estimation-based depth but thus trading off affordability for time and accuracy.

TransPose pipeline in Chapter 5 can benefit from a multi-perspective estimation setting where multiple sensors are used to capture the frame of the target from different views. Although this will likely increase the accuracy of the estimation model, the computational burden may be substantially high. This is due to the fact that two inputs will be processed simultaneously in real-time. The trade-off between speed and accuracy needs to be accounted for. Hence, a high computation facility should be in place for the estimation.

The VR - robot interfacing can be faced with serious latency problems as more features are added. The volume of data that needs to be exchanged simultaneously can be high for real-time applications depending on the processor used. An investigation can be conducted on how to compress the data and provide a more seamless communication channel while optimising the data exchange. The pipeline can also be used to investigate collaborative applications between many platforms.

Lastly, as this work targets sustainability, affordability, efficiency and automation vis-a-vis precision agriculture, factors such as availability, security, risks, laws and locations can be taken cognisance of in future developments.

References

Abdulsalam, M. and Aouf, N. (2020). Deep weed detector/classifier network for precision agriculture. In *2020 28th Mediterranean Conference on Control and Automation (MED)*, pages 1087–1092. IEEE. DOI: https://doi.org/10.1109/MED48518.2020.9183325.

Abi-Farraj, F., Pacchierotti, C., Arenz, O., Neumann, G., and Giordano, P. R. (2019). A haptic shared-control architecture for guided multi-target robotic grasping. *IEEE transactions on haptics*, 13(2):270–285. DOI: https://doi.org/10.1109/TOH.2019.2913643.

Adamides, G., Katsanos, C., Christou, G., Xenos, M., Kostaras, N., and Hadzilacos, T. (2013). Human-robot interaction in agriculture: Usability evaluation of three input devices for spraying grape clusters. *Innovation*, 24:27. DOI: https://doi.org/10.13140/2.1.1706.8483.

Adamides, G., Katsanos, C., Christou, G., Xenos, M., Papadavid, G., and Hadzilacos, T. (2014). User interface considerations for telerobotics: The case of an agricultural robot sprayer. In *Second international conference on remote sensing and geoinformation of the environment (RSCy2014)*, volume 9229, pages 541–548. SPIE. DOI: https://doi.org/10.13140/2.1.2751.1362.

Adamides, G., Katsanos, C., Parmet, Y., Christou, G., Xenos, M., Hadzilacos, T., and Edan, Y. (2017). Hri usability evaluation of interaction modes for a teleoperated agricultural robotic sprayer. *Applied ergonomics*, 62:237–246. DOI: https://doi.org/10.1016/j.apergo.2017.03.008.

Akizuki, S. and Aoki, Y. (2018). Pose alignment for different objects using affordance cues. In *2018 International Workshop on Advanced Image Technology (IWAIT)*, pages 1–3. IEEE. DOI: https://doi.org/10.1109/IWAIT.2018.8369759.

Altieri, M. A., Van Schoonhoven, A., and Doll, J. (1977). The ecological role of weeds in insect pest management systems: a review illustrated by bean (phaseolus vulgaris) cropping systems. *Pans*, 23(2):195–205. DOI: https://doi.org/10.1080/09670877709412428.

Amini, A., Periyasamy, A. S., and Behnke, S. (2021). T6d-direct: Transformers for multi-object 6d pose direct regression. In *DAGM German Conference on Pattern Recognition*, pages 530–544. Springer. DOI: https://doi.org/10.1007/978-3-030-92659-5_34.

Amrita, S. A., Abirami, E., Ankita, A., Praveena, R., and Srimeena, R. (2015). Agricultural robot for automatic ploughing and seeding. In *2015 IEEE Technological Innovation in*

ICT for Agriculture and Rural Development (TIAR), pages 17–23. IEEE. DOI: https://doi.org/10.1109/TIAR.2015.7358525.

Aronson, R. L. (2013). Farmbot: Humanity's open-source automated precision farming machine. *Technology*, 19.

Bakker, T., van Asselt, K., Bontsema, J., Müller, J., and van Straten, G. (2006). An autonomous weeding robot for organic farming. In *Field and Service Robotics*, pages 579–590. Springer. DOI: https://10.1007/11736592_48.

Balaji, B., Chennupati, S. K., Chilakalapudi, S. R. K., Katuri, R., Mareedu, K., and Scholars, R. (2018). Design of uav (drone) for crop, weather monitoring and for spraying fertilizers and pesticides. *Int J Res Trends Innov*, 3(3):42–47.

Bandala, M., West, C., Monk, S., Montazeri, A., and Taylor, C. J. (2019). Vision-based assisted tele-operation of a dual-arm hydraulically actuated robot for pipe cutting and grasping in nuclear environments. *Robotics*, 8(2):42. DOI: https://doi.org/10.3390/robotics8020042.

Bay, H., Ess, A., Tuytelaars, T., and Gool, V. (2006). "speeded-up robust features (surf). computer vision and image understanding (cviu). In *Proc of the 9th European Conference on Computer Vision. Austria: Springer*. DOI: https://doi.org/10.1007/11744023_32.

Bechar, A. and Edan, Y. (2003). Human-robot collaboration for improved target recognition of agricultural robots. *Industrial Robot: An International Journal*, 30(5):432–436. DOI: https://doi.org/10.1108/01439910310492194.

Beedu, A., Alamri, H., and Essa, I. (2022). Video based object 6d pose estimation using transformers. *arXiv preprint arXiv:2210.13540*. DOI: https://doi.org/10.48550/arXiv.2210.13540.

Bo, L., Ren, X., and Fox, D. (2014). Learning hierarchical sparse features for rgb-(d) object recognition. *The International Journal of Robotics Research*, 33(4):581–599. DOI: https://doi.org/10.1177/0278364913514283.

Bochkovskiy, A., Wang, C.-Y., and Liao, H.-Y. M. (2020). Yolov4: Optimal speed and accuracy of object detection. *arXiv preprint arXiv:2004.10934*. DOI: https://doi.org/10.48550/arXiv.2004.10934.

Bouguet, J.-Y. (2004). Camera calibration toolbox for matlab. Accessed March 2, 2020. http://www.vision.caltech.edu/bouguetj/calib_doc/index.html.

Brachmann, E., Krull, A., Michel, F., Gumhold, S., Shotton, J., and Rother, C. (2014). Learning 6d object pose estimation using 3d object coordinates. In *European conference on computer vision*, pages 536–551. Springer. DOI: https://doi.org/10.1007/978-3-319-10605-2_35.

Brachmann, E., Michel, F., Krull, A., Yang, M. Y., Gumhold, S., et al. (2016). Uncertainty-driven 6d pose estimation of objects and scenes from a single rgb image. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3364–3372. DOI: https://doi.org/10.1109/CVPR.2016.366.

Brantner, G. and Khatib, O. (2021). Controlling ocean one: Human–robot collaboration for deep-sea manipulation. *Journal of Field Robotics*, 38(1):28–51. DOI: https://doi.org/10.1002/rob.21960.

Caesar, H., Bankiti, V., Lang, A. H., Vora, S., Liong, V. E., Xu, Q., Krishnan, A., Pan, Y., Baldan, G., and Beijbom, O. (2020). nuscenes: A multimodal dataset for autonomous driving. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11621–11631. DOI: https://doi.org/10.1109/CVPR42600.2020.01164.

Calonder, M., Lepetit, V., Strecha, C., and Fua, P. (2010). Brief: Binary robust independent elementary features. In *Computer Vision–ECCV 2010: 11th European Conference on Computer Vision, Heraklion, Crete, Greece, September 5-11, 2010, Proceedings, Part IV 11*, pages 778–792. Springer. DOI: https://doi.org/10.1007/978-3-642-15561-1_56.

Cao, Z., Sheikh, Y., and Banerjee, N. K. (2016). Real-time scalable 6dof pose estimation for textureless objects. In *2016 IEEE International conference on Robotics and Automation (ICRA)*, pages 2441–2448. IEEE. DOI: https://doi.org/10.1109/ICRA.2016.7487396.

Carion, N., Massa, F., Synnaeve, G., Usunier, N., Kirillov, A., and Zagoruyko, S. (2020). End-to-end object detection with transformers. In *European conference on computer vision*, pages 213–229. Springer. DOI: https://doi.org/10.1007/978-3-030-58452-8_13.

Cerda, R., Avelino, J., Gary, C., Tixier, P., Lechevallier, E., and Allinne, C. (2017). Primary and secondary yield losses caused by pests and diseases: Assessment and modeling in coffee. *PloS one*, 12(1):e0169133. DOI: https://doi.org/10.1371/journal.pone.0169133.

Chen, H., Huang, P., and Liu, Z. (2019). Mode switching-based symmetric predictive control mechanism for networked teleoperation space robot system. *IEEE/ASME Transactions on Mechatronics*, 24(6):2706–2717. DOI: https://doi.org/10.1109/TMECH.2019.2946197.

Chen, Y., Zhang, S., Wu, Z., Yang, B., Luo, Q., and Xu, K. (2020). Review of surgical robotic systems for keyhole and endoscopic procedures: state of the art and perspectives. *Frontiers of medicine*, 14:382–403. DOI: https://doi.org/10.1007/s11684-020-0781-x.

Choi, Y., Kim, N., Hwang, S., Park, K., Yoon, J. S., An, K., and Kweon, I. S. (2018). Kaist multi-spectral day/night data set for autonomous and assisted driving. *IEEE Transactions on Intelligent Transportation Systems*, 19(3):934–948. DOI: https://doi.org/10.1109/TITS.2018.2791533.

Christoph, J. (2020). Github: Franka ros. Accessed June 12, 2021. https://github.com/frankaemika/franka_ros.

Christopher, B. (2006). *Pattern Recognition and Machine Learning*, page 227–228. New York: Springer.

Clark, J. P., Lentini, G., Barontini, F., Catalano, M. G., Bianchi, M., and O'Malley, M. K. (2019). On the role of wearable haptics for force feedback in teleimpedance control for dual-arm robotic teleoperation. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 5187–5193. IEEE. DOI: https://doi.org/10.1109/ICRA.2019.8793652.

Collet, A., Martinez, M., and Srinivasa, S. S. (2011). The moped framework: Object recognition and pose estimation for manipulation. *The international journal of robotics research*, 30(10):1284–1306. DOI: https://doi.org/10.1177/0278364911401765.

Cong, P., Zhu, X., Qiao, F., Ren, Y., Peng, X., Hou, Y., Xu, L., Yang, R., Manocha, D., and Ma, Y. (2022). Stcrowd: A multimodal dataset for pedestrian perception in crowded scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 19608–19617. DOI: https://doi.org/10.1109/CVPR52688.2022.01899.

Dai, W., Zhang, Y., Chen, S., Sun, D., and Kong, D. (2021). A multi-spectral dataset for evaluating motion estimation systems. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5560–5566. IEEE. DOI: https://doi.org/10.1109/ICRA48506.2021.9561906.

Daponte, P., De Vito, L., Glielmo, L., Iannelli, L., Liuzza, D., Picariello, F., and Silano, G. (2019). A review on the use of drones for precision agriculture. In *IOP conference series: earth and environmental science*, volume 275, page 012022. IOP Publishing. DOI: https://doi.org/10.1088/1755-1315/275/1/012022.

DataMIntelligence (2020). Agricultural drone market size, share: Industry forecast, 2027. Accessed March 7, 2022. https://datamintelligence.com/research-report/agricultural-drone-market/.

Deneault, D., Schinstock, D., and Lewis, C. (2008). Tracking ground targets with measurements obtained from a single monocular camera mounted on an unmanned aerial vehicle. In *2008 IEEE International Conference on Robotics and Automation*, pages 65–72. IEEE. DOI: https://doi.org/10.1109/ROBOT.2008.4543188.

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., et al. (2020). An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*. DOI: https://doi.org/10.48550/arXiv.2010.11929.

Doumanoglou, A., Balntas, V., Kouskouridas, R., and Kim, T.-K. (2016a). Siamese regression networks with efficient mid-level feature extraction for 3d object pose estimation. *arXiv preprint arXiv:1607.02257*. DOI: https://doi.org/10.48550/arXiv.1607.02257.

Doumanoglou, A., Kouskouridas, R., Malassiotis, S., and Kim, T.-K. (2016b). 6d object detection and next-best-view prediction in the crowd. In *CVPR*, volume 1, page 2. DOI: https://doi.org/10.48550/arXiv.1512.07506.

Dwivedi, A., Naresh, R., Kumar, R., Kumar, P., and Kumar, R. (2017). Climate smart agriculture. *December*.

Eigen, D., Puhrsch, C., and Fergus, R. (2014). Depth map prediction from a single image using a multi-scale deep network. *Advances in neural information processing systems*, 27. DOI: https://doi.org/10.48550/arXiv.1406.2283.

Engel, J., Sturm, J., and Cremers, D. (2014). Scale-aware navigation of a low-cost quadrocopter with a monocular camera. *Robotics and Autonomous Systems*, 62(11):1646–1656. DOI: https://doi.org/10.1016/j.robot.2014.03.012.

Gao, G., Lauri, M., Hu, X., Zhang, J., and Frintrop, S. (2021). Cloudaae: Learning 6d object pose regression with on-line data synthesis on point clouds. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 11081–11087. IEEE. DOI: https://doi.org/10.1109/ICRA48506.2021.9561475.

Gao, G., Lauri, M., Wang, Y., Hu, X., Zhang, J., and Frintrop, S. (2020). 6d object pose regression via supervised learning on point clouds. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3643–3649. IEEE. DOI: https://doi.org/10.1109/ICRA40945.2020.9197461.

Gazebo (2018). Gazebo simulator. Accessed May 4, 2021. https://gazebosim.org/.

Geiger, A., Lenz, P., Stiller, C., and Urtasun, R. (2013). Vision meets robotics: The kitti dataset. *The International Journal of Robotics Research*, 32(11):1231–1237. DOI: https://doi.org/10.1177/0278364913491297.

Gené-Mola, J., Vilaplana, V., Rosell-Polo, J. R., Morros, J.-R., Ruiz-Hidalgo, J., and Gregorio, E. (2019). Multi-modal deep learning for fuji apple detection using rgb-d cameras and their radiometric capabilities. *Computers and Electronics in Agriculture*, 162:689–698. DOI: https://doi.org/10.1016/j.compag.2019.05.016.

Geyer, J., Kassahun, Y., Mahmudi, M., Ricou, X., Durgesh, R., Chung, A. S., Hauswald, L., Pham, V. H., Mühlegg, M., Dorn, S., et al. (2020). A2d2: Audi autonomous driving dataset. *arXiv preprint arXiv:2004.06320*. DOI: https://doi.org/10.48550/arXiv.2004.06320.

Girbes-Juan, V., Schettino, V., Demiris, Y., and Tornero, J. (2020). Haptic and visual feedback assistance for dual-arm robot teleoperation in surface conditioning tasks. *IEEE Transactions on Haptics*, 14(1):44–56. DOI: https://doi.org/10.1109/TOH.2020.3004388.

Girbés-Juan, V., Schettino, V., Gracia, L., Solanes, J. E., Demiris, Y., and Tornero, J. (2022). Combining haptics and inertial motion capture to enhance remote control of a dual-arm robot. *Journal on Multimodal User Interfaces*, pages 1–20. DOI: https://doi.org/10.1007/s12193-021-00386-8.

Girshick, R. (2015). Fast r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 1440–1448. DOI: https://doi.org/10.48550/arXiv.1504.08083.

Girshick, R., Donahue, J., Darrell, T., and Malik, J. (2014). Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 580–587. DOI: https://doi.org/10.1109/CVPR.2014.81.

Godoy, E. P., Tabile, R. A., Pereira, R. R., Tangerino, G. T., Porto, A. J., and Inamasu, R. Y. (2010). Design and implementation of an electronic architecture for an agricultural mobile robot. DOI: https://doi.org/10.1590/S1415-43662010001100015.

Gomez-Balderas, J.-E., Flores, G., Carrillo, L. G., and Lozano, R. (2013). Tracking a ground moving target with a quadrotor using switching control. *Journal of Intelligent & Robotic Systems*, 70(1):65–78. DOI: https://doi.org/10.1007/s10846-012-9747-9.

Gonzalez-de Soto, M., Emmi, L., Perez-Ruiz, M., Aguera, J., and Gonzalez-de Santos, P. (2016). Autonomous systems for precise spraying–evaluation of a robotised patch sprayer. *Biosystems engineering*, 146:165–182. DOI: https://doi.org/10.1016/j.biosystemseng.2015.12.018.

Goodfellow, I., Bengio, Y., and Courville, A. (2016). *Deep Learning*, pages 12–14, 78, 185–191, 286–291, 298–302, 526–531. MIT Press. http://www.deeplearningbook.org.

Gorjup, G., Dwivedi, A., Elangovan, N., and Liarokapis, M. (2019). An intuitive, affordances oriented telemanipulation framework for a dual robot arm hand system: On the execution of bimanual tasks. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3611–3616. IEEE. DOI: https://doi.org/10.1109/IROS40897.2019.8967782.

Grimstad, L., Pham, C. D., Phan, H. T., and From, P. J. (2015a). On the design of a low-cost, light-weight, and highly versatile agricultural robot. In *2015 IEEE International Workshop on Advanced Robotics and its Social Impacts (ARSO)*, pages 1–6. IEEE. DOI: https://doi.org/10.1109/ARSO.2015.7428210.

Grimstad, L., Phan, H. N., Pham, C. D., Bjugstad, N., and From, P. J. (2015b). Initial field-testing of thorvald, a versatile robotic platform for agricultural applications. In *IROS Workshop on Agri-Food Robotics: dealing with natural variability*.

Guimarães, R. L., de Oliveira, A. S., Fabro, J. A., Becker, T., and Brenner, V. A. (2016). Ros navigation: Concepts and tutorial. In *Robot Operating System (ROS)*, pages 121–160. Springer. DOI: https://doi.org/10.1007/978-3-319-26054-9_6.

Guo, Z., Chai, Z., Liu, C., and Xiong, Z. (2019). A fast global method combined with local features for 6d object pose estimation. In *2019 IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM)*, pages 1–6. IEEE. DOI: https://doi.org/10.1109/AIM.2019.8868409.

Guzmán Ortiz, E., Andres, B., Fraile, F., Poler, R., and Ortiz Bas, Á. (2021). Fleet management system for mobile robots in healthcare environments. *Journal of Industrial Engineering and Management (JIEM)*, 14(1):55–71. DOI: https://doi.org/10.3926/jiem.3284.

Habibian, S., Dadvar, M., Peykari, B., Hosseini, A., Salehzadeh, M. H., Hosseini, A. H., and Najafi, F. (2021). Design and implementation of a maxi-sized mobile robot (karo) for rescue missions. *Robomech Journal*, 8(1):1–33. DOI: https://doi.org/10.1186/s40648-020-00188-9.

Hakkim, V. A., Joseph, E. A., Gokul, A. A., and Mufeedha, K. (2016). Precision farming: the future of indian agriculture. *Journal of Applied Biology and Biotechnology*, 4(6):068–072. DOI: https://doi.org/10.7324/JABB.2016.40609.

Han, L., Ruijuan, C., and Enrong, M. (2016). Design and simulation of a handling robot for bagged agricultural materials. *IFAC-PapersOnLine*, 49(16):171–176. DOI: https://doi.org/10.1016/j.ifacol.2016.10.032.

Harris, D. R. and Fuller, D. Q. (2014). Agriculture: definition and overview. *Encyclopedia of global archaeology*, pages 104–113. DOI: https://doi.org/10.1007/978-1-4419-0465-2_64.

He, K., Zhang, X., Ren, S., and Sun, J. (2016a). Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778. DOI: https://doi.org/10.1109/CVPR.2016.90.

He, K., Zhang, X., Ren, S., and Sun, J. (2016b). Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778. DOI: https://doi.org/10.1109/CVPR.2016.90.

Herrera, P. J., Dorado, J., and Ribeiro, Á. (2014). A novel approach for weed type classification based on shape descriptors and a fuzzy decision-making method. *Sensors*, 14(8):15304–15324. DOI: https://doi.org/10.3390/s140815304.

Hinterstoisser, S., Cagniart, C., Ilic, S., Sturm, P., Navab, N., Fua, P., and Lepetit, V. (2011). Gradient response maps for real-time detection of textureless objects. *IEEE transactions on pattern analysis and machine intelligence*, 34(5):876–888. DOI: https://doi.org/10.1109/TPAMI.2011.206.

Hinterstoisser, S., Lepetit, V., Ilic, S., Holzer, S., Bradski, G., Konolige, K., and Navab, N. (2012). Model based training, detection and pose estimation of texture-less 3d objects in heavily cluttered scenes. In *Asian conference on computer vision*, pages 548–562. Springer. DOI: https://doi.org/10.1007/978-3-642-37331-2_42.

Hinterstoisser, S., Lepetit, V., Ilic, S., Holzer, S., Bradski, G., Konolige, K., and Navab, N. (2013). Model based training, detection and pose estimation of texture-less 3d objects in heavily cluttered scenes. In *Asian conference on computer vision*, pages 548–562. Springer. DOI: https://doi.org/10.1007/978-3-642-37331-2_42.

Hinton, G. E., Srivastava, N., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. R. (2012). Improving neural networks by preventing co-adaptation of feature detectors. *arXiv preprint arXiv:1207.0580*. DOI: https://doi.org/10.48550/arXiv.1207.0580.

Hodan, T., Haluza, P., Obdržálek, Š., Matas, J., Lourakis, M., and Zabulis, X. (2017). T-less: An rgb-d dataset for 6d pose estimation of texture-less objects. In *2017 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 880–888. IEEE. DOI: https://doi.org/10.48550/arXiv.1701.05498.

Hodgson, J. M. (1968). *The nature, ecology, and control of Canada thistle*. Number 1386. Agricultural Research service, US Department of Agriculture.

Hosseinpoor, H., Samadzadegan, F., and Dadras Javan, F. (2016). Pricise target geolocation and tracking based on uav video imagery. *International Archives of the Photogrammetry, Remote Sensing & Spatial Information Sciences*, 41. DOI: https://doi.org/10.5194/isprsarchives-XLI-B6-243-2016.

Hou, Y. and Yu, C. (2014). Autonomous target localization using quadrotor. In *The 26th Chinese Control and Decision Conference (2014 CCDC)*, pages 864–869. IEEE. DOI: https://doi.org/10.1109/CCDC.2014.6852285.

Houston, J., Zuidhof, G., Bergamini, L., Ye, Y., Chen, L., Jain, A., Omari, S., Iglovikov, V., and Ondruska, P. (2021). One thousand and one hours: Self-driving motion prediction dataset. In *Conference on Robot Learning*, pages 409–418. PMLR. DOI: https://doi.org/10.48550/arXiv.2006.14480.

Hu, Y., Hugonot, J., Fua, P., and Salzmann, M. (2019). Segmentation-driven 6d object pose estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3385–3394. DOI: https://doi.org/10.1109/CVPR.2019.00350.

Huang, G., Liu, Z., Van Der Maaten, L., and Weinberger, K. Q. (2017). Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4700–4708. DOI: https://doi.org/10.1109/CVPR.2017.243.

Iandola, F. N., Han, S., Moskewicz, M. W., Ashraf, K., Dally, W. J., and Keutzer, K. (2016). Squeezenet: Alexnet-level accuracy with 50x fewer parameters and< 0.5 mb model size. *arXiv preprint arXiv:1602.07360*. DOI: https://doi.org/10.48550/arXiv.1602.07360.

Ioffe, S. and Szegedy, C. (2015). Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*, pages 448–456. pmlr. DOI: https://doi.org/10.48550/arXiv.1502.03167.

Ishibashi, M., Iida, M., Suguri, M., and Masuda, R. (2013). Remote monitoring of agricultural robot using web application. *IFAC Proceedings Volumes*, 46(18):138–142. DOI: https://doi.org/10.3182/20130828-2-SF-3019.00047.

Islam, N., Rashid, M. M., Wibowo, S., Xu, C.-Y., Morshed, A., Wasimi, S. A., Moore, S., and Rahman, S. M. (2021). Early weed detection using image processing and machine learning techniques in an australian chilli farm. *Agriculture*, 11(5):387. DOI: https://doi.org/10.3390/agriculture11050387.

Isop, W. A., Gebhardt, C., Nägeli, T., Fraundorfer, F., Hilliges, O., and Schmalstieg, D. (2019). High-level teleoperation system for aerial exploration of indoor environments. *Frontiers in Robotics and AI*, 6:95. DOI: https://doi.org/10.3389/frobt.2019.00095.

Jantos, T. G., Hamdad, M. A., Granig, W., Weiss, S., and Steinbrener, J. (2023). Poet: Pose estimation transformer for single-view, multi-object 6d pose estimation. In *Conference on Robot Learning*, pages 1060–1070. PMLR. DOI: https://doi.org/10.48550/arXiv.2211.14125.

Jia, X., Sun, C., and Fu, J. (2022). Mobile augmented reality centred ietm system for shipping applications. *INTERNATIONAL JOURNAL OF ROBOTICS & AUTOMATION*, 37(1):147–162. DOI: https://doi.org/10.2316/J.2022.206-0723.

Jiang, H., Zhang, C., Qiao, Y., Zhang, Z., Zhang, W., and Song, C. (2020). Cnn feature based graph convolutional network for weed and crop recognition in smart farming. *Computers and Electronics in Agriculture*, 174:105450. DOI: https://doi.org/10.1016/j.compag.2020.105450.

Kendall, A., Grimes, M., and Cipolla, R. (2015). Posenet: A convolutional network for real-time 6-dof camera relocalization. In *Proceedings of the IEEE international conference on computer vision*, pages 2938–2946. DOI: https://doi.org/10.1109/ICCV.2015.336.

Kevin, M. (2012). *Machine Learning: A Probabilistic Perspective*, pages 1–3, 9–10. MIT Press.

Khan, S., Naseer, M., Hayat, M., Zamir, S. W., Khan, F. S., and Shah, M. (2022). Transformers in vision: A survey. *ACM computing surveys (CSUR)*, 54(10s):1–41. DOI: https://doi.org/10.1145/3505244.

Kono, H., Mori, T., Ji, Y., Fujii, H., and Suzuki, T. (2019). Development of perilous environment estimation system using a teleoperated rescue robot with on-board lidar. In *2019 IEEE/SICE International Symposium on System Integration (SII)*, pages 7–10. IEEE. DOI: https://doi.org/10.1109/SII.2019.8700382.

Kot, T. and Novák, P. (2018). Application of virtual reality in teleoperation of the military mobile robotic system taros. *International journal of advanced robotic systems*, 15(1):1729881417751545. DOI: https://doi.org/10.1177/1729881417751545.

Krull, A., Brachmann, E., Michel, F., Yang, M. Y., Gumhold, S., and Rother, C. (2015). Learning analysis-by-synthesis for 6d pose estimation in rgb-d images. In *Proceedings of the IEEE international conference on computer vision*, pages 954–962. DOI: https://doi.org/10.1109/ICCV.2015.115.

Kuang, H., Liu, C., Chan, L. L. H., and Yan, H. (2018). Multi-class fruit detection based on image region selection and improved object proposals. *Neurocomputing*, 283:241–255. DOI: https://doi.org/10.1016/j.neucom.2017.12.057.

Kuznietsov, Y., Stuckler, J., and Leibe, B. (2017). Semi-supervised deep learning for monocular depth map prediction. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6647–6655. DOI: https://doi.org/10.1109/CVPR.2017.238.

Lanini, W., Strange, M., et al. (1991). Low-input management of weeds in vegetable fields. *California Agriculture*, 45(1):11–13. DOI: https://doi.org/10.3733/ca.v045n01p11.

Law, M., Ahn, H. S., Broadbent, E., Peri, K., Kerse, N., Topou, E., Gasteiger, N., and MacDonald, B. (2021). Case studies on the usability, acceptability and functionality of autonomous mobile delivery robots in real-world healthcare settings. *Intelligent Service Robotics*, 14(3):387–398. DOI: https://doi.org/10.1007/s11370-021-00368-5.

Le Cun, Y., Jackel, L. D., Boser, B., Denker, J. S., Graf, H. P., Guyon, I., Henderson, D., Howard, R. E., and Hubbard, W. (1989). Handwritten digit recognition: Applications of neural network chips and automatic learning. *IEEE Communications Magazine*, 27(11):41–46. DOI: https://doi.org/10.1007/978-3-642-76153-9_35.

LeCun, Y., Bengio, Y., and Hinton, G. (2015). Deep learning. *nature*, 521(7553):436–444. DOI: https://doi.org/10.1038/nature14539.

Lee, A. J., Cho, Y., Yoon, S., Shin, Y., and Kim, A. (2019). Vivid: Vision for visibility dataset. In *ICRA Workshop on Dataset Generation and Benchmarking of SLAM Algorithms for Robotics and VR/AR*. DOI: https://doi.org/10.48550/arXiv.2204.06183.

Lehnert, C., English, A., McCool, C., Tow, A. W., and Perez, T. (2017). Autonomous sweet pepper harvesting for protected cropping systems. *IEEE Robotics and Automation Letters*, 2(2):872–879. DOI: https://doi.org/10.1109/LRA.2017.2655622.

Lesire-Cabaniols, C. (2014). tum ardrone. Accessed May 3, 2020. https://github.com/tum-vision/tum_ardrone.

Li, P., Cai, K., Saputra, M. R. U., Dai, Z., and Lu, C. X. (2022). Odombeyondvision: An indoor multi-modal multi-platform odometry dataset beyond the visible spectrum. *arXiv preprint arXiv:2206.01589*. DOI: https://doi.org/10.1109/IROS47612.2022.9981865.

Lim, H., Kim, S.-W., Song, J.-B., and Cha, Y. (2021). Thin piezoelectric mobile robot using curved tail oscillation. *IEEE Access*, 9:145477–145485. DOI: https://doi.org/10.1109/ACCESS.2021.3122935.

Lin, T.-Y., Dollár, P., Girshick, R., He, K., Hariharan, B., and Belongie, S. (2017). Feature pyramid networks for object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2117–2125. DOI: https://doi.org/10.1109/CVPR.2017.106.

Liu, B. and Bruch, R. (2020). Weed detection for selective spraying: A review. *Current Robotics Reports*, 1(1):19–26. DOI: https://doi.org/10.1007/s43154-020-00001-w.

Liu, F., Shen, C., and Lin, G. (2015). Deep convolutional neural fields for depth estimation from a single image. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5162–5170. DOI: https://doi.org/10.1109/TPAMI.2015.2505283.

Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.-Y., and Berg, A. C. (2016). Ssd: Single shot multibox detector. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part I 14*, pages 21–37. Springer. DOI: https://doi.org/10.1007/978-3-319-46448-0_2.

Liu, X., Wang, G., Li, Y., and Ji, X. (2022). Catre: Iterative point clouds alignment for category-level object pose refinement. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part II*, pages 499–516. Springer. DOI: https://doi.org/10.1007/978-3-031-20086-1_29.

Loshchilov, I. and Hutter, F. (2017). Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*. DOI: https://doi.org/10.48550/arXiv.1711.05101.

Lottes, P., Behley, J., Milioto, A., and Stachniss, C. (2018a). Fully convolutional networks with sequential information for robust crop and weed detection in precision farming. *IEEE Robotics and Automation Letters*, 3(4):2870–2877. DOI: https://doi.org/10.1109/LRA.2018.2846289.

Lottes, P., Behley, J., Milioto, A., and Stachniss, C. (2018b). Fully convolutional networks with sequential information for robust crop and weed detection in precision farming. *IEEE Robotics and Automation Letters*, 3(4):2870–2877. DOI: https://doi.org/10.1109/LRA.2018.2846289.

Lottes, P., Khanna, R., Pfeifer, J., Siegwart, R., and Stachniss, C. (2017a). Uav-based crop and weed classification for smart farming. In *2017 IEEE international conference on robotics and automation (ICRA)*, pages 3024–3031. IEEE. DOI: https://doi.org/10.1109/ICRA.2017.7989347.

Lottes, P., Khanna, R., Pfeifer, J., Siegwart, R., and Stachniss, C. (2017b). Uav-based crop and weed classification for smart farming. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3024–3031. IEEE. DOI: https://doi.org/10.1109/ICRA.2017.7989347.

Lowe, D. G. (1999). Object recognition from local scale-invariant features. In *Proceedings of the seventh IEEE international conference on computer vision*, volume 2, pages 1150–1157. IEEE. DOI: https://doi.org/10.1109/ICCV.1999.790410.

Lowe, D. G. (2004). Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60:91–110. DOI: https://doi.org/10.1023/B:VISI.0000029664.99615.94.

Lu, Z., Huang, P., and Liu, Z. (2017). Predictive approach for sensorless bimanual teleoperation under random time delays with adaptive fuzzy control. *IEEE Transactions on Industrial Electronics*, 65(3):2439–2448. DOI: https://doi.org/10.1109/TIE.2017.2745445.

Maas, A. L., Hannun, A. Y., Ng, A. Y., et al. (2013). Rectifier nonlinearities improve neural network acoustic models. In *Proc. icml*, volume 30, page 3. Atlanta, Georgia, USA.

Mace, A., Ridley, L., Parrish, G., Barker, I., MacArthur, R., Rainford, J., Garthwaite, D., Team, S., and Hutton, S. (2017). Pesticide usage survey report 286.

Mahsa, S. (2020). Agricultural robots: Global market 2020-2025. Accessed March 7, 2022. https://statista.com/statistics/744965/agricultural-robot-global-market/.

Malneršič, A., Dular, M., Širok, B., Oberti, R., and Hočevar, M. (2016). Close-range air-assisted precision spot-spraying for robotic applications: Aerodynamics and spray coverage analysis. *Biosystems Engineering*, 146:216–226. DOI: https://doi.org/10.1016/j.biosystemseng.2016.01.001.

Marchand, E., Uchiyama, H., and Spindler, F. (2015). Pose estimation for augmented reality: a hands-on survey. *IEEE transactions on visualization and computer graphics*, 22(12):2633–2651. DOI: https://doi.org/10.1109/TVCG.2015.2513408.

Marin-Plaza, P., Hussein, A., Martin, D., and Escalera, A. d. l. (2018). Global and local path planning study in a ros-based research platform for autonomous vehicles. *Journal of Advanced Transportation*, 2018. DOI: https://doi.org/10.1155/2018/6392697.

Martin, B. (2018). Siemens/ros-sharp. Accessed May 4, 2021. https://github.com/siemens/ros-sharp.

Max, R. and Hannah, R. (2013). Hunger and undernourishment. Accessed February 7, 2023. https://ourworldindata.org/hunger-and-undernourishment.

Menze, M. and Geiger, A. (2015). Object scene flow for autonomous vehicles. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3061–3070. DOI: https://doi.org/10.1109/CVPR.2015.7298925.

Mike, P. (2020). Github: Jackal. Accessed June 12, 2021. https://github.com/jackal/jackal.

Miyake, E., Takubo, T., and Ueno, A. (2020). 3d pose estimation for the object with knowing color symbol by using correspondence grouping algorithm. In *2020 IEEE/SICE International Symposium on System Integration (SII)*, pages 960–965. IEEE. DOI: https://doi.org/10.1109/SII46433.2020.9025968.

Monaco, T., Grayson, A., and Sanders, D. (1981). Influence of four weed species on the growth, yield, and quality of direct-seeded tomatoes (lycopersicon esculentum). *Weed Science*, 29(4):394–397. DOI: https://doi.org/10.1017/S0043174500039874.

Monajjemi, M. (2014). Ardrone autonomy. Accessed May 3, 2020. https://github.com/AutonomyLab/ardrone_autonomy.

Motive (2020). Prime 13 - in depth. Accessed July 6, 2022. https://optitrack.com/cameras/primex-13/.

Mueggler, E., Faessler, M., Fontana, F., and Scaramuzza, D. (2014). Aerial-guided navigation of a ground robot among movable obstacles. In *2014 IEEE International Symposium on Safety, Security, and Rescue Robotics (2014)*, pages 1–8. IEEE. DOI: https://doi.org/10.1109/SSRR.2014.7017662.

Mueller-Sim, T., Jenkins, M., Abel, J., and Kantor, G. (2017). The robotanist: A ground-based agricultural robot for high-throughput crop phenotyping. In *2017 IEEE international conference on robotics and automation (ICRA)*, pages 3634–3639. IEEE. DOI: https://doi.org/10.1109/ICRA.2017.7989418.

Mureşan, H. and Oltean, M. (2017). Fruit recognition from images using deep learning. *arXiv preprint arXiv:1712.00580*. DOI: https://doi.org/10.48550/arXiv.1712.00580.

Nicolis, D., Palumbo, M., Zanchettin, A. M., and Rocco, P. (2018). Occlusion-free visual servoing for the shared autonomy teleoperation of dual-arm robots. *IEEE Robotics and Automation Letters*, 3(2):796–803. DOI: https://doi.org/10.1109/LRA.2018.2792143.

Niemeyer, G., Preusche, C., Stramigioli, S., and Lee, D. (2016). Telerobotics. *Springer handbook of robotics*, pages 1085–1108. DOI: https://doi.org/10.1007/978-3-319-32552-1_43.

Nowozin, S. (2014). Optimal decisions from probabilistic models: the intersection-over-union case. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 548–555. DOI: https://doi.org/10.1109/CVPR.2014.77.

Oberti, R., Marchi, M., Tirelli, P., Calcante, A., Iriti, M., Tona, E., Hočevar, M., Baur, J., Pfaff, J., Schütz, C., et al. (2016). Selective spraying of grapevines for disease control using a modular agricultural robot. *Biosystems engineering*, 146:203–215. DOI: https://doi.org/10.1016/j.biosystemseng.2015.12.004.

Oberweger, M., Rad, M., and Lepetit, V. (2018). Making deep heatmaps robust to partial occlusions for 3d object pose estimation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 119–134. DOI: https://doi.org/10.48550/arXiv.1804.03959.

Oré, G., Alcântara, M. S., Góes, J. A., Oliveira, L. P., Yepes, J., Teruel, B., Castro, V., Bins, L. S., Castro, F., Luebeck, D., et al. (2020). Crop growth monitoring with drone-borne dinsar. *Remote Sensing*, 12(4):615. DOI: https://doi.org/10.3390/rs12040615.

Paikekari, A., Ghule, V., Meshram, R., and Raskar, V. (2016). Weed detection using image processing. *International Research Journal of Engineering and Technology (IRJET)*, 3(3):1220–1222. DOI: https://doi.org/10.13140/RG.2.2.15116.64642.

Parrot (2020). Developer guide sdk 2.0. Accessed July 22, 2022. https://jpchanson.github.io/ARdrone/ParrotDevGuide.pdf.

Patil, A., Malla, S., Gang, H., and Chen, Y.-T. (2019). The h3d dataset for full-surround 3d multi-object detection and tracking in crowded urban scenes. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 9552–9557. IEEE. DOI: https://doi.org/10.48550/arXiv.1903.01568.

Pavlakos, G., Zhou, X., Chan, A., Derpanis, K. G., and Daniilidis, K. (2017). 6-dof object pose from semantic keypoints. In *2017 IEEE international conference on robotics and automation (ICRA)*, pages 2011–2018. IEEE. DOI: https://doi.org/10.1109/ICRA.2017.7989233.

Peng, S., Liu, Y., Huang, Q., Zhou, X., and Bao, H. (2019). Pvnet: Pixel-wise voting network for 6dof pose estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4561–4570. DOI: https://doi.org/10.1109/TPAMI.2020.3047388.

Ponda, S., Kolacinski, R., and Frazzoli, E. (2009). Trajectory optimization for target localization using small unmanned aerial vehicles. In *AIAA guidance, navigation, and control conference*, page 6015. DOI: https://doi.org/10.2514/6.2009-6015.

Potena, C., Khanna, R., Nieto, J., Siegwart, R., Nardi, D., and Pretto, A. (2019). Agricolmap: Aerial-ground collaborative 3d mapping for precision farming. *IEEE Robotics and Automation Letters*, 4(2):1085–1092. DOI: https://doi.org/10.1109/LRA.2019.2894468.

Pretto, A., Aravecchia, S., Burgard, W., Chebrolu, N., Dornhege, C., Falck, T., Fleckenstein, F., Fontenla, A., Imperoli, M., Khanna, R., et al. (2020). Building an aerial–ground robotics system for precision farming: an adaptable solution. *IEEE Robotics & Automation Magazine*, 28(3):29–49. DOI: https://doi.org/10.1109/MRA.2020.3012492.

Puri, V., Nayyar, A., and Raja, L. (2017). Agriculture drones: A modern breakthrough in precision agriculture. *Journal of Statistics and Management Systems*, 20(4):507–518. DOI: https://doi.org/10.1080/09720510.2017.1395171.

Pusphavalli, M. and Chandraleka, R. (2016a). Automatic weed removal system using machine vision. *International Journal of advanced Research in Electronics and Communication Engineering*, 5(3):503–506.

Pusphavalli, M. and Chandraleka, R. (2016b). Automatic weed removal system using machine vision. *International Journal of advanced Research in Electronics and Communication Engineering (IJARECE)*, 5(3):503–506.

Quigley, M., Conley, K., Gerkey, B., Faust, J., Foote, T., Leibs, J., Wheeler, R., Ng, A. Y., et al. (2009). Ros: an open-source robot operating system. In *ICRA workshop on open source software*, volume 3, page 5. Kobe, Japan.

R Shamshiri, R., Weltzien, C., Hameed, I. A., J Yule, I., E Grift, T., Balasundram, S. K., Pitonakova, L., Ahmad, D., and Chowdhary, G. (2018). Research and development in agricultural robotics: A perspective of digital farming. DOI: https://doi.org/10.25165/j.ijabe.20181104.4278.

Rad, M. and Lepetit, V. (2017). Bb8: A scalable, accurate, robust to partial occlusion method for predicting the 3d poses of challenging objects without using depth. In *Proceedings of the IEEE international conference on computer vision*, pages 3828–3836. DOI: https://doi.org/10.1109/ICCV.2017.413.

Rajmane, R., Gitay, N., Yadav, A., and Patil, A. (2020). Precision agriculture and robotics. *International Journal of Engineering Research*. DOI: https://doi.org/10.17577/IJERTV9IS010019.

Redding, J. D., McLain, T. W., Beard, R. W., and Taylor, C. N. (2006). Vision-based target localization from a fixed-wing miniature air vehicle. In *2006 American Control Conference*, pages 6–pp. IEEE. DOI: https://doi.org/10.1109/ACC.2006.1657153.

Redmon, J., Divvala, S., Girshick, R., and Farhadi, A. (2016a). You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 779–788. DOI: https://doi.org/10.48550/arXiv.1506.02640.

Redmon, J., Divvala, S., Girshick, R., and Farhadi, A. (2016b). You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 779–788. DOI: https://doi.org/10.48550/arXiv.1506.02640.

Redmon, J. and Farhadi, A. (2017a). Yolo9000: better, faster, stronger. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7263–7271. DOI: https://doi.org/10.1109/CVPR.2017.690.

Redmon, J. and Farhadi, A. (2017b). Yolo9000: better, faster, stronger. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7263–7271. DOI: https://doi.org/10.1109/CVPR.2017.690.

Ren, S., He, K., Girshick, R., and Sun, J. (2015). Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in neural information processing systems*, 28. DOI: https://doi.org/10.1109/TPAMI.2016.2577031.

Rezatofighi, H., Tsoi, N., Gwak, J., Sadeghian, A., Reid, I., and Savarese, S. (2019). Generalized intersection over union: A metric and a loss for bounding box regression. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 658–666. DOI: https://doi.org/10.1109/CVPR.2019.00075.

Rothganger, F., Lazebnik, S., Schmid, C., and Ponce, J. (2003). 3d object modeling and recognition using affine-invariant patches and multi-view spatial constraints. In *2003 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2003. Proceedings.*, volume 2, pages II–272. IEEE. DOI: https://doi.org/10.1109/CVPR.2003.1211480.

Rublee, E., Rabaud, V., Konolige, K., and Bradski, G. (2011). Orb: An efficient alternative to sift or surf. In *2011 International conference on computer vision*, pages 2564–2571. IEEE. DOI: https://doi.org/10.1109/ICCV.2011.6126544.

Ruiz, J. A. D. and Aouf, N. (2017). Unscented kalman filter for vision based target localisation with a quadrotor. *ICINCO (2)*, 2. DOI: https://doi.org/10.5220/0006474404530458.

Rumelhart, D. E., Hinton, G. E., and Williams, R. J. (1985). Learning internal representations by error propagation. Technical report, California Univ San Diego La Jolla Inst for Cognitive Science.

Sa, I., Ge, Z., Dayoub, F., Upcroft, B., Perez, T., and McCool, C. (2016). Deepfruits: A fruit detection system using deep neural networks. *sensors*, 16(8):1222. DOI: https://doi.org/10.3390/s16081222.

Saputra, R. P., Rakicevic, N., Kuder, I., Bilsdorfer, J., Gough, A., Dakin, A., de Cocker, E., Rock, S., Harpin, R., and Kormushev, P. (2021). Resqbot 2.0: an improved design of a mobile rescue robot with an inflatable neck securing device for safe casualty extraction. *Applied Sciences*, 11(12):5414. DOI: https://doi.org/10.3390/app11125414.

Saracino, A., Oude-Vrielink, T. J., Menciassi, A., Sinibaldi, E., and Mylonas, G. P. (2020). Haptic intracorporeal palpation using a cable-driven parallel robot: a user study. *IEEE Transactions on Biomedical Engineering*, 67(12):3452–3463. DOI: https://doi.org/10.1109/TBME.2020.2987646.

Satish Kumar, K. and Sudeep, C. (2007). Robots for precision agriculture. In *Electronic Proceedings of 13th National Conference on Mechanisms and Machines (NaCoMM07), Bangalore, India*, pages 12–13.

Savary, S. and Willocquet, L. (2014). Simulation modeling in botanical epidemiology and crop loss analysis. *Plant Health Instructor*, (Online):147.

Saxena, A., Sun, M., and Ng, A. Y. (2008). Make3d: Learning 3d scene structure from a single still image. *IEEE transactions on pattern analysis and machine intelligence*, 31(5):824–840. DOI: https://doi.org/10.1109/TPAMI.2008.132.

Schuster, M. J., Müller, M. G., Brunner, S. G., Lehner, H., Lehner, P., Sakagami, R., Dömel, A., Meyer, L., Vodermayer, B., Giubilato, R., et al. (2020). The arches space-analogue demonstration mission: Towards heterogeneous teams of autonomous robots for collaborative scientific sampling in planetary exploration. *IEEE Robotics and Automation Letters*, 5(4):5315–5322. DOI: https://doi.org/10.1109/LRA.2020.3007468.

Selvaggio, M., Abi-Farraj, F., Pacchierotti, C., Giordano, P. R., and Siciliano, B. (2018). Haptic-based shared-control methods for a dual-arm system. *IEEE Robotics and Automation Letters*, 3(4):4249–4256. DOI: https://doi.org/10.1109/LRA.2018.2864353.

Selvaggio, M., Moccia, R., Ficuciello, F., Siciliano, B., et al. (2019). Haptic-guided shared control for needle grasping optimization in minimally invasive robotic surgery. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3617–3623. IEEE. DOI: https://doi.org/10.1109/IROS40897.2019.8968109.

Simonyan, K. and Zisserman, A. (2014a). Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*. DOI: https://doi.org/10.48550/arXiv.1409.1556.

Simonyan, K. and Zisserman, A. (2014b). Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*. DOI: https://doi.org/10.48550/arXiv.1409.1556.

Sivčev, S., Coleman, J., Omerdić, E., Dooly, G., and Toal, D. (2018). Underwater manipulators: A review. *Ocean engineering*, 163:431–450. DOI: https://doi.org/10.1016/j.oceaneng.2018.06.018.

Solanes, J. E., Muñoz, A., Gracia, L., and Tornero, J. (2022). Virtual reality-based interface for advanced assisted mobile robot teleoperation. *Applied Sciences*, 12(12):6071. DOI: https://doi.org/10.3390/app12126071.

Srinivasan, A. (2006). *Handbook of precision agriculture: principles and applications*. CRC press.

Stuart, R. and Peter, N. (2013). *Artificial Intelligence: A Modern Approach*, pages 1–3. Number 3. Pearson Education Limited.

Su, H., Qi, C. R., Li, Y., and Guibas, L. J. (2015). Render for cnn: Viewpoint estimation in images using cnns trained with rendered 3d model views. In *Proceedings of the IEEE international conference on computer vision*, pages 2686–2694. DOI: https://doi.org/10.1109/ICCV.2015.308.

Suarez, A., Real, F., Vega, V. M., Heredia, G., Rodriguez-Castaño, A., and Ollero, A. (2020). Compliant bimanual aerial manipulation: Standard and long reach configurations. *IEEE Access*, 8:88844–88865. DOI: https://doi.org/10.1109/ACCESS.2020.2993101.

Sun, P., Kretzschmar, H., Dotiwalla, X., Chouard, A., Patnaik, V., Tsui, P., Guo, J., Zhou, Y., Chai, Y., Caine, B., et al. (2020). Scalability in perception for autonomous driving: Waymo open dataset. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 2446–2454. DOI: https://doi.org/10.48550/arXiv.1912.04838.

Sun, Z., Yang, H., Ma, Y., Wang, X., Mo, Y., Li, H., and Jiang, Z. (2021). Bit-dmr: A humanoid dual-arm mobile robot for complex rescue operations. *IEEE Robotics and Automation Letters*, 7(2):802–809. DOI: https://doi.org/10.1109/LRA.2021.3131379.

Sundermeyer, M., Marton, Z.-C., Durner, M., Brucker, M., and Triebel, R. (2018). Implicit 3d orientation learning for 6d object detection from rgb images. In *Proceedings of the european conference on computer vision (ECCV)*, pages 699–715. DOI: https://doi.org/10.48550/arXiv.1902.01275.

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., and Rabinovich, A. (2015). Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9. DOI: https://doi.org/10.48550/arXiv.1409.4842.

Tang, J.-L., Chen, X.-Q., Miao, R.-H., and Wang, D. (2016). Weed detection using image processing under different illumination for site-specific areas spraying. *Computers and Electronics in Agriculture*, 122:103–111. DOI: https://doi.org/10.1016/j.compag.2015.12.016.

Technologies, U. (n.d). Unity. Accessed February 1, 2023. https://unity.com/.

Tejani, A., Tang, D., Kouskouridas, R., and Kim, T.-K. (2014). Latent-class hough forests for 3d object detection and pose estimation. In *European Conference on Computer Vision*, pages 462–477. Springer. DOI: https://doi.org/10.1007/978-3-319-10599-4_30.

Tekin, B., Sinha, S. N., and Fua, P. (2018). Real-time seamless single shot 6d object pose prediction. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 292–301. DOI: https://doi.org/10.1109/CVPR.2018.00038.

Thapa, P. (2021). Potential of unmanned aerial vehicles for agriculture: A review. *Review of Behavioral Aspect in Organizations and Society*, 3:1–8. DOI: https://doi.org/10.32770/rbaos.vol31-8.

Tian, Y., Duan, H., Luo, R., Zhang, Y., Jia, W., Lian, J., Zheng, Y., Ruan, C., and Li, C. (2019). Fast recognition and location of target fruit based on depth information. *IEEE Access*, 7:170553–170563. DOI: https://doi.org/10.1109/ACCESS.2019.2955566.

Tokekar, P., Vander Hook, J., Mulla, D., and Isler, V. (2016). Sensor planning for a symbiotic uav and ugv system for precision agriculture. *IEEE Transactions on Robotics*, 32(6):1498–1511. DOI: https://doi.org/10.1109/TRO.2016.2603528.

Touvron, H., Cord, M., Douze, M., Massa, F., Sablayrolles, A., and Jégou, H. (2021). Training data-efficient image transformers & distillation through attention. In *International Conference on Machine Learning*, pages 10347–10357. PMLR. DOI: https://doi.org/10.48550/arXiv.2012.12877.

Tu, S., Xue, Y., Zheng, C., Qi, Y., Wan, H., and Mao, L. (2018). Detection of passion fruits and maturity classification using red-green-blue depth images. *Biosystems Engineering*, 175:156–167. DOI: https://doi.org/10.1016/j.biosystemseng.2018.09.004.

Tulsiani, S. and Malik, J. (2015). Viewpoints and keypoints. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1510–1519. DOI: https://doi.org/10.1109/CVPR.2015.7298758.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. (2017). Attention is all you need. *Advances in neural information processing systems*, 30. DOI: https://doi.org/10.48550/arXiv.1706.03762.

Vive, H. (n.d). Vive pro 2 full kit. Accessed February 1, 2023. https://www.vive.com/uk/product/vive-pro2-full-kit/overview.

Wan, E. A. and Van Der Merwe, R. (2000). The unscented kalman filter for nonlinear estimation. In *Proceedings of the IEEE 2000 Adaptive Systems for Signal Processing, Communications, and Control Symposium (Cat. No. 00EX373)*, pages 153–158. Ieee. DOI: https://doi.org/10.1109/ASSPCC.2000.882463.

Wang, A., Zhang, W., and Wei, X. (2019a). A review on weed detection using ground-based machine vision and image processing techniques. *Computers and electronics in agriculture*, 158:226–240. DOI: https://doi.org/10.1016/j.compag.2019.02.005.

Wang, C., Xu, D., Zhu, Y., Martín-Martín, R., Lu, C., Fei-Fei, L., and Savarese, S. (2019b). Densefusion: 6d object pose estimation by iterative dense fusion. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 3343–3352. DOI: https://doi.org/10.1109/CVPR.2019.00346.

Wang, J., Chen, K., Xu, R., Liu, Z., Loy, C. C., and Lin, D. (2019c). Carafe: Content-aware reassembly of features. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 3007–3016. DOI: https://doi.org/10.1109/ICCV.2019.00310.

Wang, Z., Walsh, K. B., and Verma, B. (2017). On-tree mango fruit size estimation using rgb-d images. *Sensors*, 17(12):2738. DOI: https://doi.org/10.3390/s17122738.

Wei, X., Jia, K., Lan, J., Li, Y., Zeng, Y., and Wang, C. (2014). Automatic method of fruit object extraction under complex agricultural background for vision system of fruit picking robot. *Optik*, 125(19):5684–5689. DOI: https://doi.org/10.1016/j.ijleo.2014.07.001.

Wohlhart, P. and Lepetit, V. (2015). Learning descriptors for object recognition and 3d pose estimation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3109–3118. DOI: https://doi.org/10.1109/CVPR.2015.7298930.

Xiang, Y., Schmidt, T., Narayanan, V., and Fox, D. (2017). Posecnn: A convolutional neural network for 6d object pose estimation in cluttered scenes. *arXiv preprint arXiv:1711.00199*. DOI: https://doi.org/10.48550/arXiv.1711.00199.

Xie, Z., Singh, A., Uang, J., Narayan, K. S., and Abbeel, P. (2013). Multimodal blending for high-accuracy instance recognition. In *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2214–2221. IEEE. DOI: https://doi.org/10.1109/IROS.2013.6696666.

Yin, K., Sun, Q., Gao, F., and Zhou, S. (2022). Lunar surface soft-landing analysis of a novel six-legged mobile lander with repetitive landing capacity. *Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science*, 236(2):1214–1233. DOI: https://doi.org/10.1177/0954406221999422.

Yoon, H.-S., Jeong, J. H., and Yi, B.-J. (2018). Image-guided dual master–slave robotic system for maxillary sinus surgery. *IEEE Transactions on Robotics*, 34(4):1098–1111. DOI: https://doi.org/10.1109/TRO.2018.2830334.

Yu, H., Fu, Q., Yang, Z., Tan, L., Sun, W., and Sun, M. (2018). Robust robot pose estimation for challenging scenes with an rgb-d camera. *IEEE Sensors Journal*, 19(6):2217–2229. DOI: https://doi.org/10.1109/JSEN.2018.2884321.

Yu, J., Schumann, A. W., Cao, Z., Sharpe, S. M., and Boyd, N. S. (2019a). Weed detection in perennial ryegrass with deep learning convolutional neural network. *Frontiers in plant science*, 10:1422. DOI: https://doi.org/10.3389/fpls.2019.01422.

Yu, J., Schumann, A. W., Cao, Z., Sharpe, S. M., and Boyd, N. S. (2019b). Weed detection in perennial ryegrass with deep learning convolutional neural network. *frontiers in Plant Science*, 10:1422. DOI: https://doi.org/10.3389/fpls.2019.01422.

Zarukin, D. (2014). Github: oneapi-src/onednn. Accessed May 4, 2021. https://github.com/oneapi-src/oneDNN.

Zeiler, M. D. and Fergus, R. (2014). Visualizing and understanding convolutional networks. In *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part I 13*, pages 818–833. Springer. DOI: https://doi.org/10.48550/arXiv.1311.2901.

Zhang, X., Jiang, Z., Zhang, H., and Wei, Q. (2018). Vision-based pose estimation for textureless space objects by contour points matching. *IEEE Transactions on Aerospace and Electronic Systems*, 54(5):2342–2355. DOI: https://doi.org/10.1109/TAES.2018.2815879.

Zhang, X., Zhu, Y., Li, C., Zhao, J., and Li, G. (2014). Sift algorithm-based 3d pose estimation of femur. *Bio-medical materials and engineering*, 24(6):2847–2855. DOI: https://doi.org/10.3233/BME-141103.

Zheng, Y.-Y., Kong, J.-L., Jin, X.-B., Wang, X.-Y., Su, T.-L., and Zuo, M. (2019a). Cropdeep: The crop vision dataset for deep-learning-based classification and detection in precision agriculture. *Sensors*, 19(5):1058. DOI: https://doi.org/10.3390/s19051058.

Zheng, Y.-Y., Kong, J.-L., Jin, X.-B., Wang, X.-Y., Su, T.-L., and Zuo, M. (2019b). Cropdeep: The crop vision dataset for deep-learning-based classification and detection in precision agriculture. *Sensors*, 19(5):1058. DOI: https://doi.org/10.3390/s19051058.

Zhu, M., Derpanis, K. G., Yang, Y., Brahmbhatt, S., Zhang, M., Phillips, C., Lecce, M., and Daniilidis, K. (2014). Single image 3d object detection and pose estimation for grasping. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3936–3943. IEEE. DOI: https://doi.org/10.1109/ICRA.2014.6907430.

Zhu, S., Yang, T., Mendieta, M., and Chen, C. (2020). A3d: Adaptive 3d networks for video action recognition. *arXiv preprint arXiv:2011.12384*. DOI: https://doi.org/10.48550/arXiv.2011.12384.