Building Agriculture AI System with Data Fusion and Computer Vision

By

Wei Zhao

A dissertation submitted in partial fulfillment of
the requirements for the degree of

Doctor of Philosophy
(Biological Systems Engineering)

at the
UNIVERSITY OF WISCONSIN-MADISON
2020

Date of final oral examination: 12/03/2020

The dissertation is approved by the following members of the Final Oral Committee:
Troy Runge, Associate Professor, Department of Biological Systems Engineering
Brian Luck, Assistant Professor, Department of Biological Systems Engineering
Matthew Digman, Assistant Professor, Department of Biological Systems Engineering
Zhou Zhang, Assistant Professor, Department of Biological Systems Engineering
Cong Chen, Research Associate Faculty, University of South Florida

ACKNOWLEDGMENT

I want to say thank you to all the individuals accompanying me along the way in my Ph.D. life.

I have to say that I am a lucky guy, a lucky student who met such a wonderful advisor, Professor Troy Runge. I had other graduate degrees before, which I thought I already spent all my luck on meeting good advisors. But I was wrong, my good luck kept going on and on. Because I had the chance to know, learn, and enjoy my second Ph.D. life with Professor Troy Runge – more like a friend than an advisor. Thanks for your endless support, caring, inspiration, and guidance throughout my Ph.D. study. Thanks for letting me enjoy my research more than ever.

Special thanks to my thesis committee members, Professor Matthew Digman, Professor Zhou Zhang, Professor Brian Luck, and Dr. Cong Chen, for their suggestions, critiques, and help shaping the direction of this dissertation. Professor Matthew Digman offered great support both in constructing my research theory and conducting my wide experiments. Professor Zhou Zhang gave my confidence to persistently continue my study on combining AI and traditional agriculture. Professor Brian Luck is always the right person to discuss and validate my wild research ideas ensuring my research's feasibility. Dr. Chen is my power resource in transferring my previous research result and skillset to the new ability in augmenting my knowledge.

I am fortunate to work with many talented colleagues during my Ph.D. Zong Liu brought me to this research area. Bozhao Qi and Lei Kang helped me in diving deep into AI related skillsets. William Yamada kept contributing to my experiments. I am grateful for the help and support from my fellow graduate students, including Tianxin Li, Xuan Wang, Peng Liu, Tao Ji, Haoran Qiu, Shengzhi He, Xuan Zhang, Lei Kang and many others.

Lastly but most importantly, I would like to express my sincere gratitude to my family – my wife and my two little daughters. My wife tool most workload for the family and took care of the kids well. And it is amazing that my kids barely trouble me when I was focusing on my research. Without their continuous support, constant love and encouragement, none of this would have been possible without the love of my family. This dissertation is dedicated to every one of them!

To my wife and kids

Contents

LIST OF FIGURES

LIST OF TABLES

ABSTRACT

Smart farming concepts have made significant contributions to increase food production and sustainability in the 21st century. These systems combine software and hardware that allows farmers to measure, monitor and control certain agronomic parameters. Smart farming technologies are becoming more prominent with technological advances in farming in fields of automation, ICT (information and communication technology), and robotics. Several applications of artificial intelligence (AI) are already developed for use in agriculture and farming purposes and it is anticipated that use of "AI farming" will assist the agriculture sector in the future to drive more efficient production.

To understand how to leverage the advantages of smart farming system in developing innovative farming applications, the following questions need to be answered: (1) What kinds of sensors should be used and data collected considering the cost and efficiency? (2) How do we seamlessly transfer new AI technologies from other industries to agricultural problems which have different requirements in terms of accuracy, scalability, operation, environment, etc. (3) What customized and innovative method should be developed dedicated to agriculture on the top of current general AI technologies? and (4) How should we leverage the advantages of smart farming?

Motivated by these questions, this research focuses on creating a modular farm mapping system. This system presents work regarding three specific studies: (i) Fundamentally creating advanced field mapping and navigation algorithms based on computer vision techniques. (ii) Profitably augmenting the mapping module by a sensing approach with a variety of sensors and

data fusion. (iii) Precisely augmenting agriculture objects perceiving ability for the map using transfer learning and computer vision techniques.

The automated mapping and navigation system investigated in study 1 could be a cornerstone of most autonomous agricultural system. Accordingly, we propose a ground-level mapping and navigating system based on computer vision technology (Mesh Simultaneous Localization and Mapping algorithm, Mesh-SLAM) and Internet of Things (IoT), to generate a 3D farm map on both the edge side and cloud. Our evaluation indicates that: 1) this Mesh-SLAM algorithm outperforms in mapping and localization precision, accuracy, and yield prediction error (resolution at centimeter); and 2) The scalability and flexibility of the IoT architecture make the system modularized, easy adding/removing new functional modules or IoT sensors. We conclude the trade-off between cost and performance widely augments the feasibility and practical implementation of this system in real farms.

In study 2 we present a sensing algorithm, a low-cost, robot-mounted, multidimensional map augmentation method that can track robot movements, monitor the surrounding environment, and link all the factors to the 3D map, thereby providing useful analytics to task planning, route planning and robot operators. The method leverages IMU sensors to gather mobility data for every individual robot. The ability to detect obstacles allows us to further augment the insight of the mapping method as a 4D or even higher dimension map rather easily. In this chapter, we attempted to provide analytics and data fusion from several specific aspects of the robot working environment. We believe that our farmland sensing approach has many more interesting and useful applications in similar agriculture environments.

In study 3, we further develop computer vision-based crop detection with unmanned aerial vehicle (UAV) acquired images. This is a critical tool for precision agriculture, but object detection using deep learning algorithms rely on a significant amount of manually prelabeled training datasets as ground truths. Field object detection such as bales is especially difficult because of 1) long-period images acquisitions under different illumination conditions and seasons, 2) limited existing prelabeled data, and 3) few pretrained model and research as reference. This work augments the bale detection accuracy only using limited data collection and labeling, by building an innovative algorithms pipeline. From the work we can conclude the algorithm pipeline improves the bale detecting performance including the recall, mAP, and F1 score with diverse illumination, seasons, and weather conditions images. This approach has strong scalability on many other crops and field objects and will significantly enable precision agriculture techniques.

Combining all the created systems, we construct an agriculture AI system with multiple innovative algorithms. This large scope of system and pieces of algorithms fill in needed gaps for creating maps to enable smart agriculture, while also providing a valuable dataset and algorithms for future researchers.

Chapter 1 Introduction

1.1 Motivation

1.1.1 Population Growing

The conflict between global population growing and low efficiency of food production: In recent decades agriculture systems have faced both agriculture labor shortage due to the nature of the work and the need to increase production with minimized inputs. The trend is expected to continue with climate change and increasing population further adding stress to these systems. The U.S. National Agricultural Statistics show that the number of farms and ranches has decreased by 3% from 2012 to 2017, and the land for agriculture has decreased by 2% (14.3 million acres) [1]. This decrease occurred while the United States population increased by 11.2 million from 2012 to 2017 [2], with similarly changes occurring globally [3]. Other stresses on agricultural production such as drought, political issues, pandemics, such as the recent COVID-19 outbreak, can also cause worldwide intermittent shortage of farm products. Additionally, agronomic producers face growing concerns of the high cost of management, limited ability of crop monitoring, pressures to minimize environmental impact.

1.1.2 Requirements of AI Technology and Digital Workforce

Seeding and Planting: Precision agriculture technologies have developed yield monitoring maps, seeding and soil maps to improve farming profits as crop yield shows a curvilineal result when the density of seeding is under-optimized. This is an emerging area of research with several recent research studies that link precision agriculture and crop yield data together. For example, Massey et al. [4] built a yield data map with the several years' GPS data from a farm in Missouri. They also overlaid the real cost in farm-level when they transferred the yield maps into profitability maps. Breaking a large field into segments using an index like poor topsoil routinely fails may be a way to lower seeding rates, decrease chemical use, or change the conservation uses. Additionally, Van Raij et al. [5] showed another evidence that soil maps built based on guidance systems can alter fertilizer applications with a good fitting for soil's characteristics. A study [6] by the United States Department of Agriculture demonstrated that leveraging results of applying precision agriculture technologies based on combining both farm level practice data and operator financial data from a nationally representative sample of corn farms. The conclusions suggested there could be cost saving in seeding and planting operation by demonstrating the cost versus the yield gains on different-sized of fields. Also, with future development of precision agriculture technologies, it is likely that farmers will be able to further increase profits and environmental benefits.

Site-specific and chemical application: With the help of precision agriculture (site-specific applications), farmers can take advantage of localized data about the soil status, growing status, and other site-specific data to optimize the management of the farm. Evert et al. [7] showed in their study that crop spectral reflectance made be used as a vegetation index to determine crop health and amount. The development of site-specific variable-rate devices – precise spreaders, sprayers, on-board rate controllers, etc. – make the field work easier than before. Basso et al. [8] created a simulation approach that can conduct quantify studies on N-leaching and field yields under various environment, chemical usage and soil conditions. This method helps manage N-fertilizer-rate related to precipitation-based water availability and radiation. The other study by Basso et al. [9] also showed the advantage of SALUS-model on economics and environment when using site-specific fertilizer applications on segmented field with year as data collection unit. Besides the studies around using N-fertilizer to enhance crop yield, recent studies have also considered the environmental impact related to N-fertilization, including nitrate leaching and nitrous oxide emissions. This precision agriculture technique can offer insights to management strategy on crops [10] under various environments and soil conditions.

Real-Time Monitoring: A number of research studies have investigated the application of computer vision in different key steps in agriculture, including observing crop growing, detecting diseases, and facilitating crop harvest [11]. Computer vision techniques have been used to collect nutritional status of plants. For example, Romualdo et al. [12] conducted a research on maize plants to diagnose plant’s nitrogen nutritional status by implementing computer vision technique at different development stages. Compared to the traditional method which relies on human observations, the computer vision technique improves the detection efficiency and accuracy. Another study by Pérez-Zavala et al. [13] proposed a computer vision approach to detect the grape bunches in vineyard scenes relying on the shape, texture descriptors and bunch separation strategy to realize automatic monitoring of grapevine growth.

With the reduction of equipment costs, increase of computing power, and availability of non-destructive food assessment methods, the efforts of many researchers and practitioners to improve the crop quality and yields have focused on computer vision and machining learning [14]. Computer vision helps with object detection and machine learning allows useful information that can be extracted from the collected data to be available, showing tremendous advantages over the traditional methods applied in agriculture [15].

Several research efforts have shown that the combination of computer vision and machine learning techniques on the multiple periods of crop production and harvesting are promising [16]. Computer vision in agriculture can be applied easily to analyze digital images collected from the fields and to provide high-level understandable information to the users [17]. For example, computer vision not only detects the weeds fast and effortlessly, but also accurately applies treatment with the help of ground robot [18]. In addition, computer vision can detect the diseases on the crops and inform users to take action [19, 20].

During the harvest process, the logistics of biomass aggregation and transportation is essential. For example, the United States has significant lignocellulosic biomass [21] resources that could be used for emerging industries like biofuels. However, converting the biomass to renewable energy is not currently economically feasible with more efficient collection methods needed [22]. With the application of computer vision and machine learning, baled biomass can be detected quickly and accurately, benefiting the harvest process by allowing for improved collection routes and yield determination. Although this is just one example it demonstrates how this technology can play a significant role in the crop harvest process.

Image Acquisitions: To collect the images as inputs for the computer vision, using Unmanned Aerial Vehicle (UAV) is an efficient approach, which has been widely used in precision agriculture as well as many other fields, such as path planning, design, and livestock monitoring [23,24]. UAV combined with computer vision can also contribute to remote sensing to help inform farmers about the geo-specific crop yield and identify crop diseases [25, 26]. Sometimes, decisions are required to be made off-board once the data have been collected and processed by the UAV, based on the information provided by the images processed from the computer vision technique [27]. For example, UAVs can be used to detect a potential issue, and then obtain high-resolution images or inspect and apply treatments correspondingly.

Disease detection: Computer vision techniques can help with disease detection in agriculture. Oberti et al. [28] implemented computer vision to detect powdery mildew on grapevine leaves and the accuracy has been improved significantly by adjusting the view angles to 40 to 60 degrees and hence improve the overall quality of the plants. Pourreza et al. [29] explored the application of computer vision technique to detect Huanglongbing disease on trees infected by a citrus psyllid. To analyze the performance of the model, laboratory and field experiments were taken and the results showed that the new method improve the target disease detection accuracy to 95.5% to 98.5%. Instead of identifying single disease, computer vision technique also contributes to the classification of multiple diseases of crops. Maharlooei et al. [30] applied image processing technology on detecting and counting soybean aphids to achieve the identification and enumeration of mites with lower costs and high accuracy in the strong light condition. Toseef and Khan [31] used fuzzy inference system to generate an intelligent mobile application to help rural farmers diagnose diseases commonly occur on wheat and cotton crops with a 99% accuracy, reducing the loss of farmers due to crop diseases and dramatically improve the crop yields.

Crop harvest: is another aspect that benefits from the computer vision techniques. Barnea et al. [32] developed crop harvesting robots by using color-agnostic shape-based 3D fruit detection technique on registered image and depth to address the localization issue in precision agriculture due to shape variations and occlusions. Lehnert et al. [33] designed an approach based on effective vision algorithms for harvesting sweet pepper and protect the cropping system, which was demonstrated to be successive by the experiments of harvesting sweet peppers from modified crop and unmodified crop.

How biomass is managed after crop harvesting is essential to balance economic and environmental impacts. Residual biomass does not have to be removed after harvesting, rather it can be incorporated into the ground to increase soil carbon. However, it can be removed in some situations with the amount taken out impacting the economics and field fertility. Biomass collection can provide economic and in certain cases may also benefit future crops. Residual biomass collected from crop fields are usually baled to a compact form before collection and transportation. In addition, stacking the bales to utilize the efficient bale-hauling equipment is sometimes desired. There are some other benefits of forming the bales into stacks. These include efficiently clearing the crop field for next grow cycle, avoiding bales troubling crop management operations, and shortening the time costs between harvest and planting.

1.2 AI Technology Aided Solution

A potential solution to mitigate some of the issues outlined in the previous section are autonomous and precision agricultural systems. The International Society of Precision Agriculture [34] adopted the following definition of precision agriculture in 2019: “Precision Agriculture is a management strategy that gathers, processes and analyzes temporal, spatial and individual data and combines it with other information to support management decisions according to estimated variability for improved resource use efficiency, productivity, quality, profitability and sustainability of agricultural production.” Precision agriculture includes precise irrigation quantity, correct and appropriate application of chemicals, weed cleaning, where topological characteristics of the crop field is a key component needs to be considered. For instance, quantity of irrigation varies drastically due to the change of terrain slops. Hussnain et al. [35] pointed out that low areas of crop field are likely to collect more water from either the irrigation and rainfall, meaning that irrigation for those areas are better to be less compared to other areas with high slopes. Mareeles et al. [36] concluded that precision agriculture in terms of irrigation system relies heavily on the topological characteristics of the crop field.

An autonomous system is build based on precise maps created with path, terrain, crops, and other objects. Crop monitoring is also essential to allow robots to distinguish the crop from weeds, monitor plant health, and determine crop maturity. Computer vision implemented via low-cost visual sensors provides strong support for both local navigation and crop monitoring. However, there are certain related technical challenges in rural fields including data transmission with high bandwidth and high speed, system scalability in different sizes of land, mapping and localization accuracy, updating and maintenance, etc. Rapid advancements in computer vision, mapping, and the Internet of Things (IoT) have provided some solutions as follows.

A guidance system capable of being scalable to the spatial range of the agriculture applications, specifically large farmlands, is the key step to achieve high agricultural efficiency. A three-dimensional (3D) navigation system, which can guide a robot autonomously, is a necessary step to enable plant monitoring. Thus, mobile robotics should have precious information about their position and be connected with the other robotics via IoT architecture.

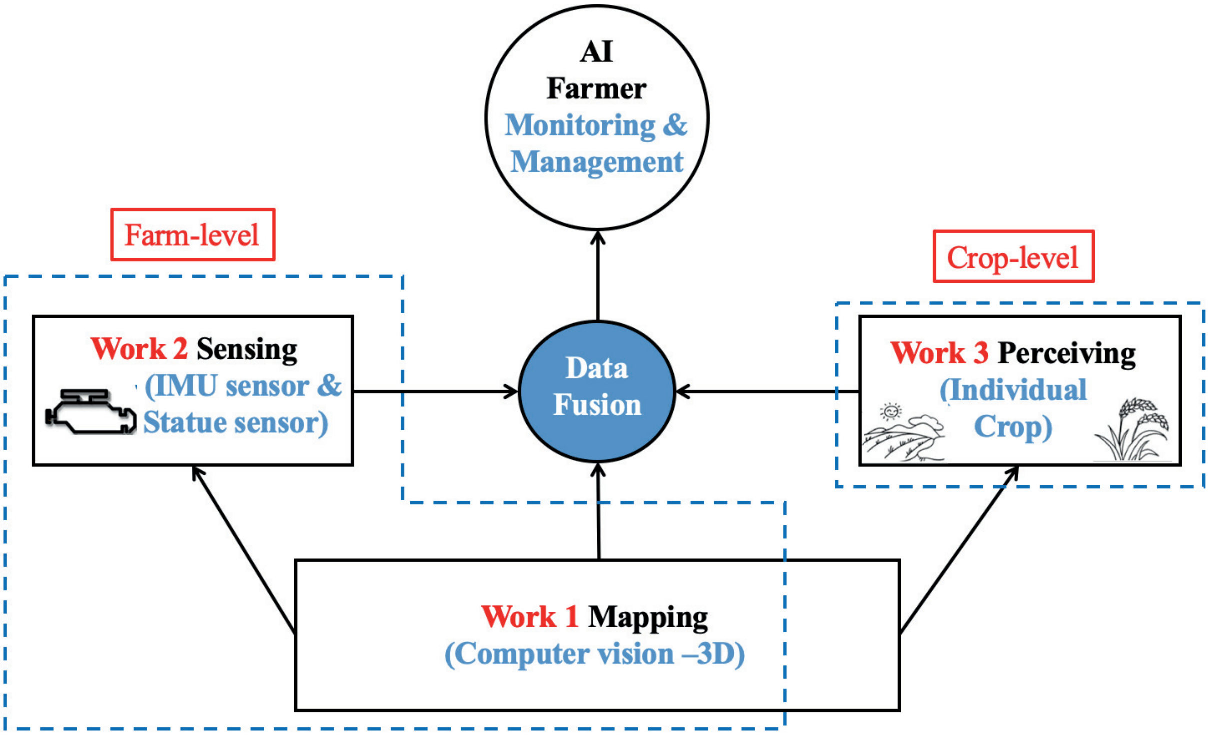


Figure 1 The architecture of the completed system

These systems are hoped to reduce labor issues for the most dangerous and tedious agronomic tasks, improve efficiency, and reduce environmental impacts through better utilization of crop inputs. Mapping, sensing, and precise perception are key technologies for enabling autonomous navigation systems which in turn will enable agricultural AI system. To lay a solid foundation, a high-level overview of the system design could be divided into three modules as is shown in Fig. 1, in which the mapping system and the sensing approach are based on farm level. These two approaches could be applied to any farm with different landscapes because of the comprehensive dimensions' information. The perceiving algorithm identifies single plants, which also could be deployed in different fields with common agriculture objects.

1.3 Related AI Technologies for Precision Agriculture

1.3.1 Mapping Approach in Agriculture

Computer vision methods have been highly involved in automated plant monitoring approaches, with representative approaches summarized in Table 1. More recent approaches have utilized ground-level image data over overhead distant images from satellites or UAVs.

3D-Mapping for farms Geometrically mapping between scenarios with changing visual features is a significant step of 3D-Map reconstruction. This data association has been recently utilized in other studies including developing a technique to map varying scenarios by key visual features in different seasons [41]; work done to provide mapping with high robustness under illumination and seasonal variation using scene recognition and localization [42], [43]; a spatial-temporal map that was highly robust to seasonal variations [44]; and a LIDAR system that was adopted to obtain the information in a vertical direction [45]. But these approaches are highly dependent on the prior information of the plant shape, which constrains them from a wide application.

Table 1 Some of the representative approaches

Temporal categories	**Approach**	**Strength**	**Weakness**
Early approach	Satellite imagery [37]	Large landscape coverage	Costly, low special and temporal resolution
Recent approaches	Unmanned Aerial Vehicles (UAVs) [38]	The capability of collecting big data of high special and temporal resolution	Traditional SfM and MVS failed to process dynamic scenes, e.g. growing plants.
	Inexpensive image sensors [39]	Scanning plants and make estimations with computer vision techniques	
	Multi-View Stereo (MVS) [40]	The capability of getting condensed and fine-grained 3D reconstruction	

Smart-Farming with IoT Farm data are increasingly being collected and with the data increasing in size. This collection of farm data to make decision leaks to the concept which we is called Smart Farming, where farming parctices are more data-driven and data-enabled. This new concept of Smart Farming [46] is the outcome of the rapid development of the Internet of Things (IoT) and cloud computing services. Smart Farming is surpassing precision agriculture because it is depending on both the location and data, improved by environment consciousness, and prompted by real-time instances [47]. It is vital to enhance the spatial farmland surveillance capacity to enlarge the agriculture productivity. Tse-Chuan et. al. [48] presented a creative IoT agriculture platform leveraging cloud and fog computing. With the help of fog and cloud, the proposed system can be applied to large-area data collection and analysis, allowing farmland with limited network information resources to be integrated and automated with agricultural monitoring automation, and other related analysis in large areas. Existing work also outlines the challenges and constraints when deploying the IoT in the domain of food and agriculture [49], [50]. Plant monitoring is a key step in navigation where a robot is guided safely and autonomously even in an unknown environment. Thus, mobile robotics should have precise information about their position and be connected with the other robotics via IoT architecture. Muangprathub [51] monitored temperature, humidity, and soil moisture over a large area using wireless sensor networks. Jirapond et at. [52] applied the concept of IoTs with wireless sensor networks to observe the moisture condition of the crop soil as well as other key factors reflecting the crop growth condition, such as and smartly controlled the quantity of watering as needed. According to the observation, the system smartly waters the crop to control the water resources and maintains the field product. In addition to monitor plant growth, IoT sensors are also deployed in the dairy industries for animal health monitoring and analysis.

Mohit et al. [53] built a platform named SmartHerd based on the IoT application to monitor dairy farming and analyze animal behaviors. The system overcame the constrained internet accessibility of remote farm by using a fog computing paradigm. The case study with a 6-month field deployment resulted in an 84% reduction in data transferred to the cloud. Mohit et al. [54] extended their previous platform to leverage the fog computing paradigm for the purpose of smart dairy farming. The system has been tested in the field where has a full dairy herd of 150 cows. The results showed that early lameness detection could be achieve before the existence of visual sign with 87% accuracy by implementing blended clustering and classification machine learning methods. Smart farming can also contribute to the protection of environmentally sensitive regions adversely impacted by farming animals. Tim et al. [55] developed a system based on wireless sensor networks to form a virtual fencing to constraint cattle movement in a certain area to avoid the interference with adjacent protected lands. Different from physical fencing method, this system is more easily to be implemented and cost efficient.

1.3.2 Advanced Sensors and Data Fusion in Agriculture

Adverse topological characteristics detection in crop field for the precision agriculture is another key opportunity. Large machines for crop watering, applying pesticide to control diseases, and harvesting are commonly used for scaling crop production. When the terrain is uneven, large autonomous machines need a detailed topography map, otherwise they could become stuck in the crop fields which would cause economic loss from loss in time or field damage. If the topography is extreme, lack of proper map could risk damage to the machinery, as navigated routes could cause machinery rollovers on steep slopes.

Advanced sensors and computer vision techniques provide an opportunity to collect topological characteristics from the environment required by precision agriculture. Sensors are becoming smaller and more powerful (Nandurkar et al. [56]). In addition, as the costs of sensors decrease, a greater deployment in practice is enabled. The development of computer vision technique is rapidly reaching the level that users can efficiently process collected data for decision making.

However, collecting all information from the crop field not only increase the burden of memory needed by the equipment, but also add more useless computing tasks. A method to extract critical and useful information from the field is needed. We propose a method that combines the advanced sensors and IMU as well as the algorithm to monitor and crop field in real time at a low cost and in a practical manner. In addition, the automated monitor system also performs at field analysis of the relationship between the topological characteristics and crop growth to provide useful information for farmers to adjust their crop management strategies accordingly and timely.

Remote monitoring Agriculture industries traditionally implements optical and multispectral techniques to the images collected by satellites to analyze and evaluate the plant growth condition and yields. The concentration of chlorophyll, for instance, could be revealed by the light absorption from the leaf and hence determines the plant health. It is of importance since decisions, such as fertilizing the soil and spreading insecticide or fungicide, should be consider the health of the plant. The treatments should be applied in time to ensure their functionality, requiring the filed information to be collected and analyzed frequently, which challenges the traditional method. In addition, traditional monitoring methods are expensive to implement regarding time efficiency.

Motion sensor With the advancement of precise GPS equipment and cameras detecting structures, various autonomous machines are employed in the agriculture sector [57]. The QUAD-AV project tested the performance of using microwave radar, stereo vision, LiDAR, and thermography to detect obstacles in the context of agriculture [58]. The project concluded that stereo vision outstands for its accuracy of ground and non-ground classification. Zhao et al. [59] proposes a method to gather road information on a large scale through the combination of phase cameras and motion sensors. An environment map was developed by [60] for detecting the relative positions of nearby objects with multiple phone cameras. In addition, the turning movement of vehicles is captured by comparing the centrifugal force with a reference point. Beall et al. [60] extended this work by designing an efficient algorithm to reduce the computing time. Non-vision sensors are utilized in Hergt's study [61] to realize the detection of vehicle maneuvers, including lane changing, turning, and moving on a curvy path. Abrams et al. [62] proposes to utilize the front and rear-mounted cameras in a phone to identify dangerous road conditions. Estes et al. [63] develops a multi-view 3D network to detect objects on the road, combined with a sensory-fusion framework to analyze data collected by LIDAR and cameras.

Mobile phones have been applied for driving safety in recent research. In the past study, experimental vehicles with mobile phone sensors are used to record driving data or test for potential crashes. Smartphone sensors data is commonly used to collect and evaluate the traffic prediction model and estimate the freeway traffic status [64]. A mobile device was also used to obtain vehicle location information and process received data. Then the route information was shared with a mobile phone to test the traffic-monitoring system [65]. Dai et al. [66] developed a technique to detect the driving behavior of drunk drivers. Driving performance was used by the proposed system to judge whether the drivers were intoxicated. Mantouka et al. [67] also collected driving motion data using sensors embedded in mobile phones. A two-stage clustering approach is developed for detecting dangers events, acceleration profile, and speeding. A larger samples experiment including 100 drivers' IMU sensor data was explained in [68]. This research took advantage of both smartphone sensors and back-end server, exploring the performance of the model on predicting phone usage in different road-types/length trips. Furthermore, Kanarachos [69] and Kang [70] argued the existing drawbacks of using smartphone sensors inters of sensors hardware, experiment methods, and data analytic algorithms. Cybernetics model and machine learning methods were applied to enhance the capability of precise driving monitoring.

Vehicle motion detection is a primary focus of transportation research, with most of the concepts and methods based on high-end sensors. Multiple external sensors, like microphones, accelerometers, and radios, are used to detect the motion and status of traffic. The driving status associated with crashes are analyzed with real-time trip data to recognize a potential accident [71]. Microsoft has designed a system to detect traffic honking, road bumps, and brakes with external sensors [72]. Several studies have used external accelerometers [73,74] to detecting potholes and other road surfaces and conditions. A recent study by Dang-Nhac et al. [75] detected driving activity and driving events through a new approach to optimize data window size and overlapping ratio for every single vehicle for training model purpose.

As a conclusion, motion sensors are wildly and deeply applied in a lot of real-world scenarios like autonomous vehicles and smartphones with mature fundamental technologies. These methods are great supplements to improve the common GPS performance. However, the implementations of motion sensors are still at primary level with simple sensor data analytic methods. The advantages of motion sensors include low energy cost, low price, portability to all kind of hardware platforms and accuracy. These advantages improve the feasibility for agriculture machines tracking and localization.

Robots applied in agriculture Robots are gaining interests in production agriculture due to the potential capability to automatically remove weeds and minimize the usage of pesticides and herbicides in crop production. Different from the traditional methods that treats the whole field uniformly, robots apply resources to the target plants individually and therefore improve the resource efficiency. Thus datasets that focus on providing field data to those who design automatic systems with robots should be very helpful to plan activities, such as navigation and mapping in modern agriculture.

Sensor data fusion in agriculture Fusion of sensor data has enormous potential for agriculture with the need to consider multiple sensor data from the environment. Fusion of GPS and machine vision is leading the way to improve the applications of machine guidance systems in agriculture. Plants cultivated in patterns facilitates the usage of autonomous machine systems with satisfying accuracy. However, adverse environments exist in agricultural applications, where the environment (hills or trees) might disable the signal of GPS negatively impact the system efficiency. Acceleration data provided by the IMU devices compensate the impact of signal loss of GPS and ensure the accuracy of guidance systems. Therefore, the combination of GPS and IMU sensors should be considered.

1.3.3 Computer Vision-based Agriculture Object Detection

Transfer Learning and Domain Adaptation. Transfer learning is a popular machine learning technique that aims to help with repetitive tasks by using an existing model. When it comes to situations where labelled data are only available in a source domain, Domain Adaption (DA), is a common technique to transfer learning as shown in Fig. 2. A small change or domain shift, due to illumination, pose, and image quality, between the source and target domains can lead to the degrading performance of machine learning models. Domain adaption (DA) provides an opportunity to mimic the human vision system which allows it to perform new tasks in a target domain by using the labelled data from a or more relevant source domains. Several research studies have recently addressed the issue of domain shift [76-80].

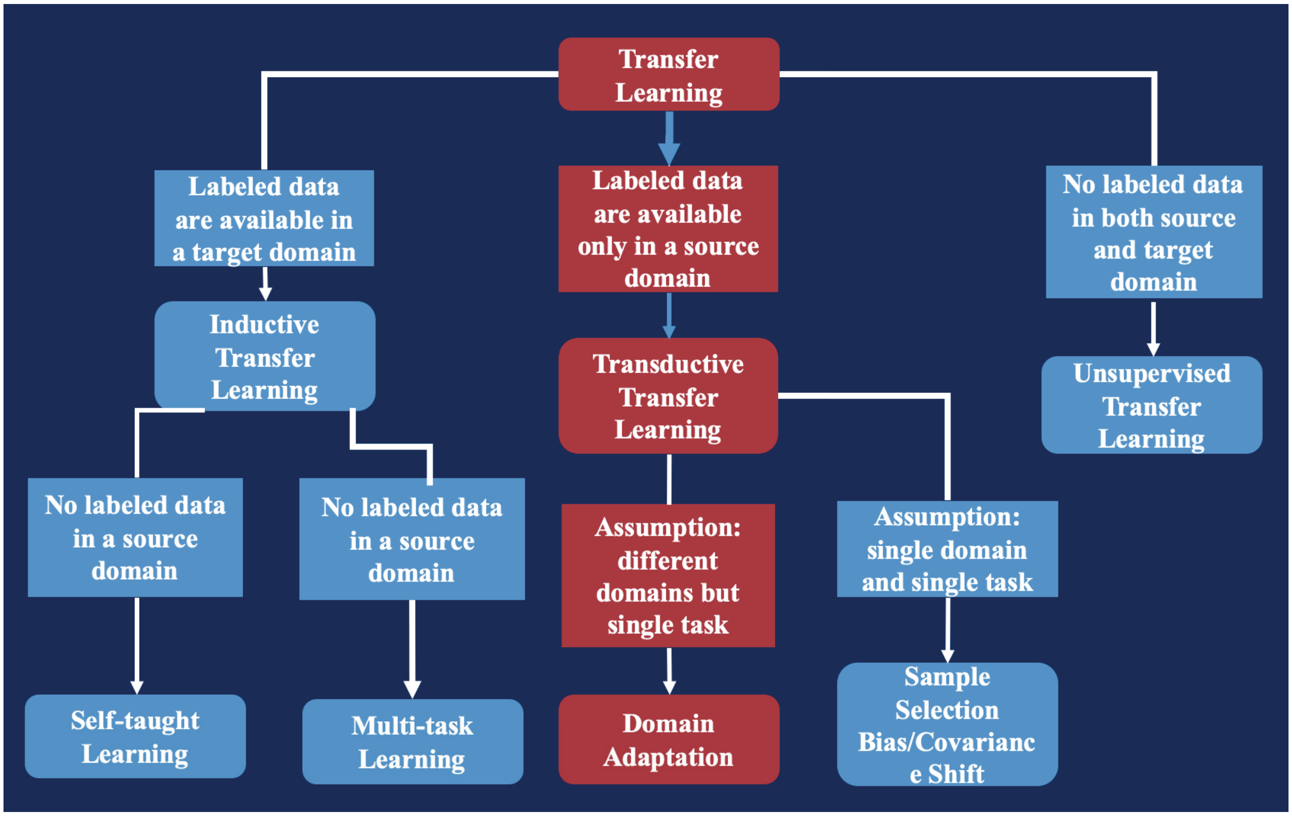

Transfer Learning
Labeled data are available in a target domain
Labeled data are available only in a source domain
No labeled data in both source and target domain
Inductive Transfer Learning
Transductive Transfer Learning
Unsupervised Transfer Learning
No labeled data in a source domain
No labeled data in a source domain
Assumption: different domains but single task
Assumption: single domain and single task
Self-taught Learning
Multi-task Learning
Domain Adaptation
Sample Selection Bias/Covariance Shift

Figure 2 Relationship between Transfer learning and Domain Adaptation

To implement CNN techniques, a large images dataset with manually labeled targets are required, which is expensive and challenging [76]. However, synthesizing images through use of the DA techniques can reduce the images need to be collected from the field and solve the problem when the labeled data cannot be acquired from the target domain [77]. Various research studies have been conducted on this concept, and have achieved promising results. Othman et al. [78] designed a domain adaption network to overcome the issues of domain shift in classification scenarios where the labeled images from the source domain and unlabeled ones from the target have different geographical features. Overall, when it comes to the problems of domain shift between source and target domains, the DA technique can not only reduce the costs of data preparation, but also improve image recognition [79, 80].

1.3.4 AI Techniques from Other Industries

Note: This section was published in my publication "Augmenting self-driving with remote control: Challenges and directions." and “Real-Time Vehicle Motion Detection and Motion Altering for Connected Vehicle: Algorithm Design and Practical Applications” before finishing the thesis.

AI techniques are widely applied in many industries. In which, self-driving or autonomous vehicle has a significant connection with autonomous agriculture since the overlap of the core techniques between these two areas. Autonomous vehicle systems are being designed over the world with increasing success in recent years. This development makes autonomous farm system be realizable.

A self-driving vehicle is one that is capable of sensing its environment and navigating itself without human input [81]. It uses a variety of techniques to sense its surroundings, such as LIDAR, RADAR, odometry, and computer vision. It uses these different sensor inputs to understand its environment, recognize various road conditions, traffic lights, road signs, lane boundaries, and track surrounding vehicles. The potential benefits of self-driving vehicles include increased safety, increased mobility and lower costs. It is estimated that self-driving vehicles can reduce 90% of the accidents and prevent up to $190 billion in damages and health-costs annually [82]. Many commercial and academic endeavors are putting significant resources for the development and tests of such self-driving systems [83, 84, 85]. For example, Google started its self-driving project in 2009, and has spent more than $1 billion in building and testing fully self-driving vehicles [86]. While legal and political challenges remain in its widespread adoption, there are also some technical bottlenecks on the way of developing completely reliable self-driving systems. All self-driving systems make decisions based on the perception of the environment and predefined traffic rules. However, there has been occasional failures of these systems when they have encountered scenarios that were hitherto unseen. For instance, based on the situation in a construction zone, human drivers would realize that it is permissible to cross over a double yellow line by following the appropriately placed cones (which otherwise is illegal to cross in the US), while a self-driving vehicle may not be able to do so, and therefore be unable to move forward. Similarly, in poor weather conditions or due to traffic light malfunctions, the cues from different sensors may contradict each other leading to confusion in decision making.

Self-Driving Systems: Corporations like Waymo, Mercedes-Benz and AutoX are trying to develop fully self-driving vehicles [83, 84, 85]. Waymo uses LIDAR as the primary input for object detection [85]. AutoX proposes camera-first self-driving solution to reduce the cost to build a self-driving vehicle [83]. [87] presents a sensory-fusion perception framework that combines LIDAR point cloud and RGB images as input and predicts oriented 3D bounding boxes. Leonard et al. [88] describes the architecture and implementation of an autonomous vehicle designed to navigate using locally perceived information in preference to potentially inaccurate or incomplete map data. Eun-Kyu et al. [89] presents networked self-driving vehicles to coordinate and form an edge computing platform. We believe remote control system can act as a safe backup for such self-driving systems.

Network technology: Reducing network latency is an active area of research. Ankit et al. [83] investigates the causes of latency inflation in the Internet and proposes a grand challenge for the networking research community: a speed-of-light Internet. Mamta and Zhang [91, 92] propose various architectures and techniques for high capacity and low latency 5th generation mobile networks. Stonebraker et al. [93] discusses the requirements of system design for real-time streaming. Song et al. [94] presents a Wi-Fi based roadside hotspot network to operate at vehicular speeds with meter-sized picocells. Kyungmin et al. [95] uses speculation to predict future frames to reduce latency for mobile cloud gaming. Huang et al. [96] develops a passive measurement tool to study the inefficiency in today's LTE networks. Najah et al. [97] presents the features to improve quality of service in LTE networks. Tu et al. [98] presents the inefficiencies of current VoLTE architectures. These studies can inspire the design of remote-control systems for self-driving vehicles.

Another commonly seen AI technique is the advanced usage of Smartphone sensors for autonomous vehicle. Smartphone sensors data is commonly used to collect and evaluate the traffic prediction model and estimate the freeway traffic status [99]. A mobile device was also used to obtain vehicle location information and process received data. Then the route information was shared with a mobile phone in order to test the traffic-monitoring system [100]. A study [101] developed a technique to detect the driving behavior of drunk drivers. Driving performance was used by the proposed system to judge whether the drivers were intoxicated. Mantouka et al. [102] also collected driving motion data using sensors embedded in mobile phones. A two-stage clustering approach is developed for detecting dangers events, acceleration profile, and speeding. A larger samples experiment including 100 drivers' IMU sensor data was explained [103]. This research took advantage of both smartphone sensors and back-end server, exploring the performance of the model on predicting phone usage in different road-types/length trips. Furthermore, the existing drawbacks of using smartphone sensors inters of sensors hardware, experiment methods, and data analytic algorithms are argued in [104,105]. Cybernetics model and machine learning methods were applied to enhance the capability of precise driving monitoring.

Vehicle motion detection is always the focus of transportation research. Most of the conceptive methods are based on high-end sensors. Multiple external sensors, like microphone, accelerometer, and radio, are demonstrated to detect the motion and status of traffic. The driving status associated with crash is analyzed with real-time trip data to recognize a potential accident [106]. Microsoft has designed a system to detect traffic honking, road bumps, and brakes with external sensors [107]. With the help of external accelerometers, studies [108,109] developed pothole patrol system for detecting road conditions, e.g. recognizing bumpy road surfaces. Dang-Nhac et al. [95] focus on driving activity detection and driving events recognition via addressing a new approach to optimize data window size and overlapping ratio for every single vehicle for training model purpose.

Various techniques and methods of detecting overtaking have been researched. Hao and Zhu [110,111] have promoted a system that used GPS and phones to detect acceleration and deceleration to estimate the congestion. A mixed algorithm was created to detect the acceleration combining dynamic planning with robust information [112]. A wireless sensor networks layout was designed to monitor vehicles [113]. Moreover, the future motion was predicted using dynamic and kinematic models making certain control inputs, vehicle capabilities, and external situation related to the updating status of vehicles [114].

Any **connected safety applications** are less meaningful without widely applied in most vehicles. However, full-scaled vehicle motion detection is a challenging task and a long-term issue in mixed traffic of automated and manual vehicles. First, there is hardly a common standard device that was approved as an accurate detector. Second, as a result of diverse car manufactories and cost of communication devices, it will take a long time to make an agreement on the popularization of the same model device. For example, the most common devices are loop detectors, magnetic sensors, acoustic sensors, and computer vision techniques. However, these techniques require special hardware to be installed either on the infrastructures or in vehicles. This also limits the wide application and scalability because of the high cost [115].

1.4 Contribution

To accomplish a comprehensive task with a agriculture robot, a major challenge is to acquire enough context information, e.g. 3D map, landscape information, and the perception of agriculture objects. Current mapping solutions can either accurately build a partial map using high-cost devices or by constructing a 2D map with a UAV (unmanned aerial vehicle). These methods fail to provide a comprehensive mapping method that is low-cost, general, accurate, and extensible at the same time for collecting both the plants and the surrounding context information. The chapters of this thesis, provide solutions to these issues and are organized as follows:

1. **A Mesh-mapping algorithm**

This chapter describes a vision-based mapping algorithm involving edge computing to overcome the difficulties faced by the current methods as is shown in Fig. 3. It is precise, inexpensive, and mobile-robot-friendly in agriculture scenarios. It leverages high computation-force edge nodes that supply the Wi-Fi access points (Aps) to users and provides computation power to localization algorithms. This scheme works out the problem that a user device doesn't have sufficient computation power to do visual-based tasks. Also, it solves the problem that a centralized server fails to support large quantities of concurrent robots. The edge nodes are managed by a cloud server. And this design has two advantages. One is that confidential information could be filtered by edge nodes before uploading. Another one is that multiple nodes could be regulated by the cloud server to implement navigation among multiple nodes. The study proposes the weakness of SLAM can be solved to become robust to environment variations by deep network methods.

(1) A precise and expandable image-based mapping and navigation algorithm which contains edge nodes and a cloud server.

(2) A mesh-based method to convert a SLAM map into a proper coordinate for implementation convenience.

(3) Flexibility on any plant ground shape but not limited to row-shaped plants. The map could adapt to any shaped area.

(4) An IoT framework to keep sufficient frame rate data are sent to edge node using UDP protocol. Reduce the feature data size by applying Each Node layer image processes before transmitting data to cloud server.

2. Advanced sensors usage and data fusion algorithm

IMU sensors [115] (a gyroscope, and an accelerometer for measuring the attitude) were used as the main tool for data collection. Many existing solutions for motion tracking and gesture recognition rely on the IMU sensors built-in mobile devices [116]. Magnetic sensor [117] readings were used to determine device orientation changes, and future applied to detect the detailed environment. Additional frames [118] are captured by the camera mounting in the front of the robot for future analytics.

The use of multiple sensors, such as a gyroscope [119] and an accelerometer [120], is advantageous because the low cost and easy combination, which makes it possible to collect continuous motion data, implement a detection model for land surface condition, and link to an individual location in the map built as described using the methodology in chapter 3. Additionally, to guarantee the accuracy of land surface condition detection, a data fusion mechanism [121] was developed that ensure the algorithm to offer accurately extended information to our SLAM map.

3. Augmenting agriculture object detection ability with transfer learning

This study tested a transfer learning method by collecting the data from the fields by the UAV equipped with RGB cameras, including 243 captured with good illimitation conditions along with 150 images in other conditions. Manually labeling each baled biomass from these conditional images was used to train the YOLOv3 model.

This study also applied the traditional background subtraction algorithm developed in [122] with the same data. The results show that our method gained the best F scores, indicating that it performs well when dealing with the discrepancy of domain distribution due to the different outdoor environments. Part of the images was manually labeled, while the rest of the images with different illumination contexts share the same labels by implementing CycleGAN for domain transferring. The processed images were used as inputs for our YOLOv3 model to perform bale detection. The goal was to show that our model, a combination of computer vision and domain adaption, could improve the accuracy and efficiency of baled detection. The essential contributions of this part are listed in 3 key points:

(1) For bale detection under illumination condition, a YOLOv3 model was built. The associate training dataset will be released with current work to fill the empty of the bale training dataset with labels as ground truths.

(2) We constructed an innovative object detection approach (algorithms pipeline) including YOLOv3 and Domain Adaptation (DA). Additionally, this approach improves the capability of bale detection.

(3) We augmented the labeled training data with more scenarios using Domain Adaptation. Combining with our manually labeled data, we are able to provide a valuable training dataset of over 1000 bale images.

Chapter 2 Mapping: A Computer Vision-based Ground-Level Mapping

This chapter has been accepted by the journal IEEE ACCESS entitled “Ground-level Mapping and Navigating for Agriculture based on IoT and Computer Vision.”

2.1 Introduction

An agricultural system robot navigation system is implemented by path planning on maps created that monitors both terrain, crops, and other objects. Crop monitoring is also essential to allow robots to distinguish between crops and weeds, monitor plant health, and determine crop maturity. Computer vision on inexpensive visual sensors provides strong support for both local navigation and crop monitoring. However, there are certain related technical challenges in rural fields including data transmission with high bandwidth and high speed, system scalability in different sizes of land, mapping and localization accuracy, updating and maintenance, etc. Rapid advancements in computer vision, mapping, and the Internet of Things (IoT) have provided some solutions as follows.

Being scalable of the spatial range of the agriculture applications, e.g. large farmlands, is the key step to achieve high agricultural efficiency. A 3D reconstruction-based navigation, where a robot is localized and guided autonomously with secure even in an unknown scenario, is the significant step of plant monitoring. So, mobile robotics should have precious information about their position and be connected with the other robotics via IoT architecture.

2.2 Theoretical Background

2.2.1 SLAM

SLAM is a method to reconstruct the environment in three dimensions and track the movement of the sensor in the environment [123]. The sensors could be inertial measurement unit (IMU), RGB cameras, LIDAR, or GPS [124]. Visual SLAM (vSLAM) only relies on visual data, e.g. photos and depth, and has been a hot topic for a while. It requires three inputs: monocular, RGBD, and stereo, with the solution performed in one of two ways. The first is a feature-based solution, where the inconsistency of image features in sequential image streams is used to recognize camera movement, for example, Mono SLAM [125], PTAM [126] and ORB-SLAM [127]. ORB-SLAM is the most recent technique with reported 1% error of map dimensions. A second solution is a direct method which takes all the images as a unity, as described in DTAM [128] and LSD-SLAM [129]. The SLAM methods are ideal for applications that use smart devices. DTAM requires GPU to become real-time, ORB-SLAM and LSD-SLAM require CPU. PTAM could be real-time on mobile phones if the map is not large [126]. In agricultural systems the outdoor navigation is for a larger area, with the paper looking at a large outdoor field. Thus, the state-of-art SLAM methods would be taxing to the CPU and the device power supply.

2.2.2 IoT and Edge Computing

In edge computing, the tasks are run at the edge of the network [130], which differs from previous systems where the computation work is done by centralized cloud servers. But after the evolution in IoT techniques, the data size has increased tremendously and data transmission and processing have become more challenging. If the computation is finished at the edges and data is kept locally, there are less delay, higher throughput and more confidential [131]. In Para-Drop [132], Wi-Fi routers are treated as edge nodes that directly communicate with users. However, there are very few edge computing applications through a lot of research effort has been reported [133]. One example is using edge computing to performing video streaming [134], and another example is applying edge computing to process big datasets on smart electronic grids [135]. This paper would be another in-detailed contribution to this area.

2.2.3 Challenge

Considering the precise and performance of the mapping algorithm, the computer vision method and cameras are the best options. A mono-camera tracking algorithm could achieve real-time and accurate performance on a normal laptop with no GPU. For instance, ORB-SLAM reports an error of 1% [136]. Also, the camera orientation is built in the SLAM output. This avoids the difficulty of sensor binding. So, SLAM fits properly in the farm localization application.

However, there are several unavoidable challenges to utilize SLAM into agricultural navigation:

(1) The CPUs of smart devices are not as strong as laptops, which could easily fail from the SLAM computation load.

(2) Power consumption of smart devices.

(3) Mobile robots do not have enough storage for the maps which are usually larger than thousands of megabytes.

(4) Download times and battery life of mobile robots.

If the SLAM is performed on a centralized cloud server, the computing power and the network bandwidth is challenged if tremendous concurrent robots are doing the image streaming request. Also, the SLAM is designed with a static environment, so it would be hard since the plants change visually along with time.

2.3 Methodology

2.3.1 Initialization

SLAM map: Typically, localization and mapping are processed concurrently in SLAM. Here, pre-constructed maps are used to re-localize the camera as shown in Fig. 5. These maps are built by higher-accuracy devices, like stereo cameras, and they are uploaded to edge nodes. With the assumption that switching between edge nodes is performed in the low-level mechanism when the device is roaming, and the switching is opaque to users, the map scope allocated to an edge node is designed to be bigger compared with the designed region to guarantee a seamless edge node switching in the situation that the next edge node is not connected yet while the user is already outside of the map coverage of the original node.

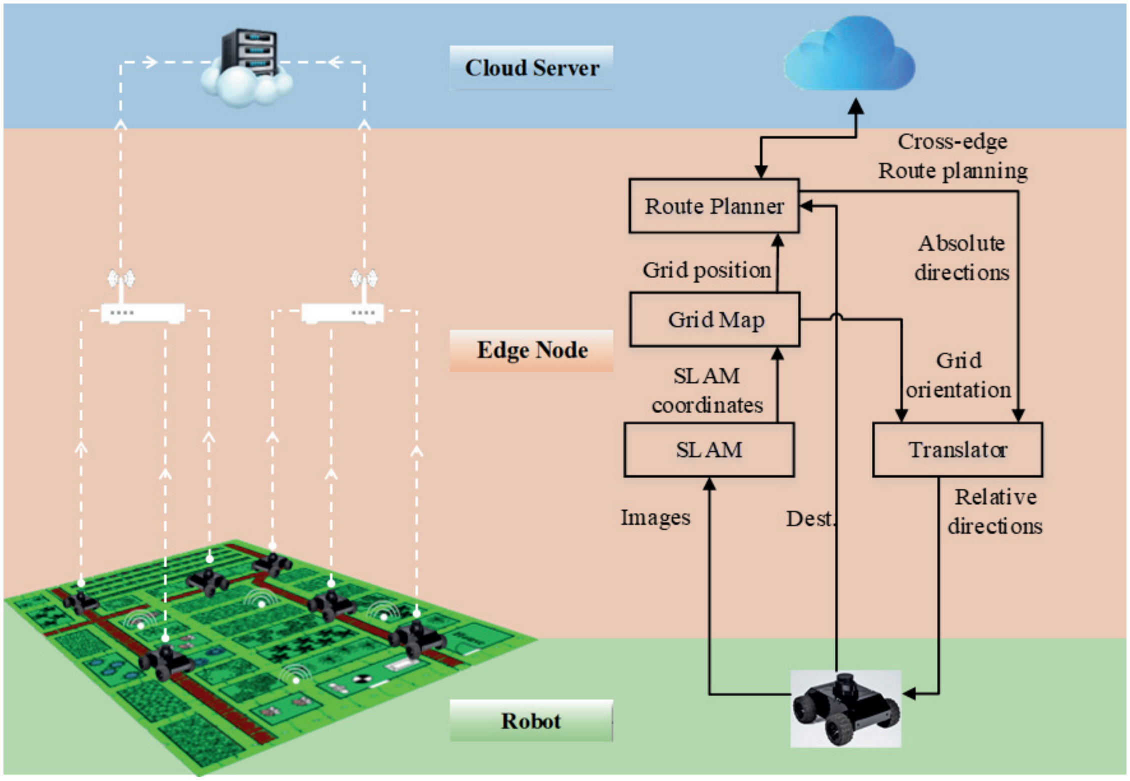


Figure 3 The IoT architecture of Cloud-Edge-Robot

Meshing information: The alignment between the SLAM map and the mesh map is implemented by mapping the horizontal and vertical unit vectors of the mesh algorithm into the SLAM coordinates. The horizontal and vertical axis correspond to the column and the row respectively, while their product is pointing in the upward normal direction. By deducting the SLAM result of one camera location, and together with m and n, the number of columns and rows in the mesh, are easily obtained.

Destinations: In the process of constructing the map or performing re-location afterward, the information of the destinations and their corresponding places in SLAM is recorded by a pair, as shown in Fig.5-a. is the SLAM coordinates and is a unique string key. The pair will be transformed into where is the corresponding mesh coordinator. These pairs are stored in a hash table for lookup in the future.

2.3.2 mn-Scaled Meshing

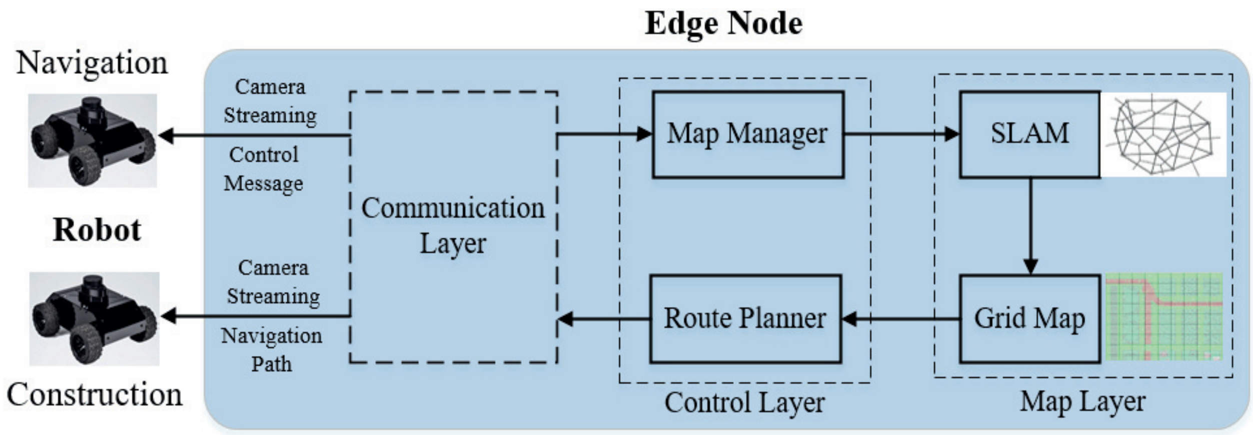


Figure 4 The Structure of the Edge Node Layer

SLAM algorithms typically are used for localization and motion tracking. Thus this use would be limited in navigational maps. Only the keyframes and the features with large intensity gradients in three-dimensional Euclidean space are recorded in a SLAM map [137]. The real-world scale can’t be reflected by the map built by a monocular camera, but prior information of the landscape, like barriers, feasible routes, and plants are stored for real-life mapping. In our implementation, seamless inter-edge navigation was provided between neighboring edge nodes by the navigation map [138], which is shown in Fig. 4.

A mesh map expanding the horizontal farm landplane is especially proposed to serve the requirements as shown in Fig. 5.

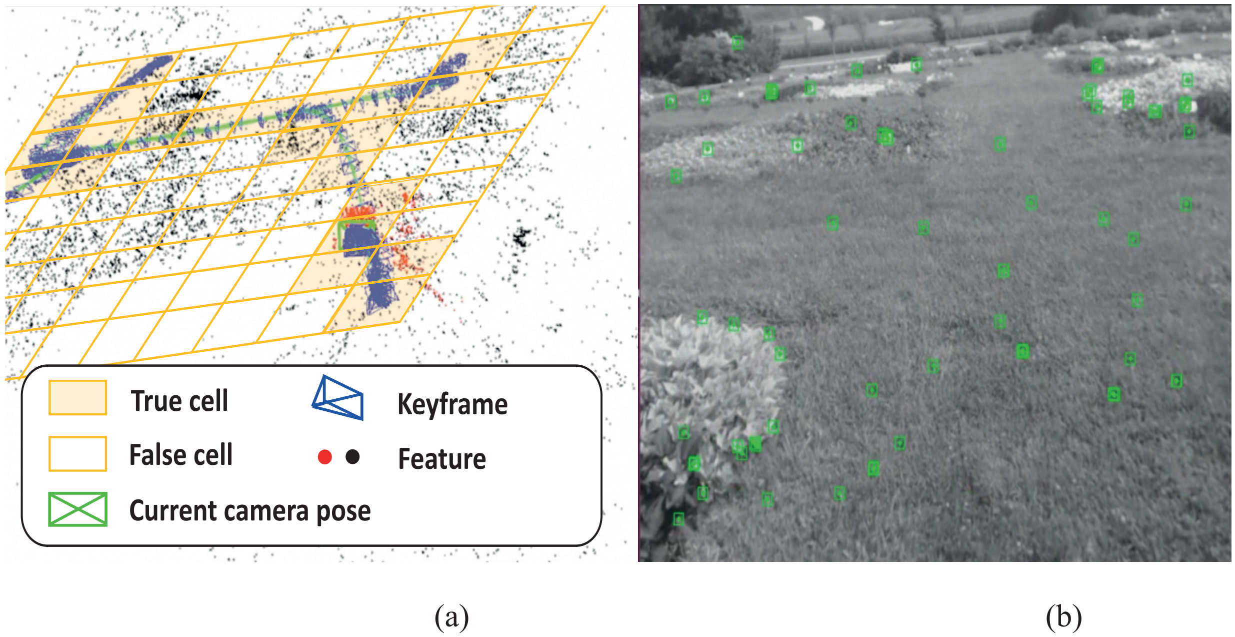


Figure 5 a) A demonstration of mn-scaled meshing with SLAM map and b) real-time farm view

This is implemented by an mn-scaled meshing algorithm with each unit of the mesh taking a Boolean value to show whether the corresponding area is accessible for robots. Keyframes from SLAM are used to complete the matrix. Besides, this mesh algorithm is based on three assumptions:

(1) The land is continuous with no terrace.

(2) The keyframes are captured at a vertical height range with small variance.

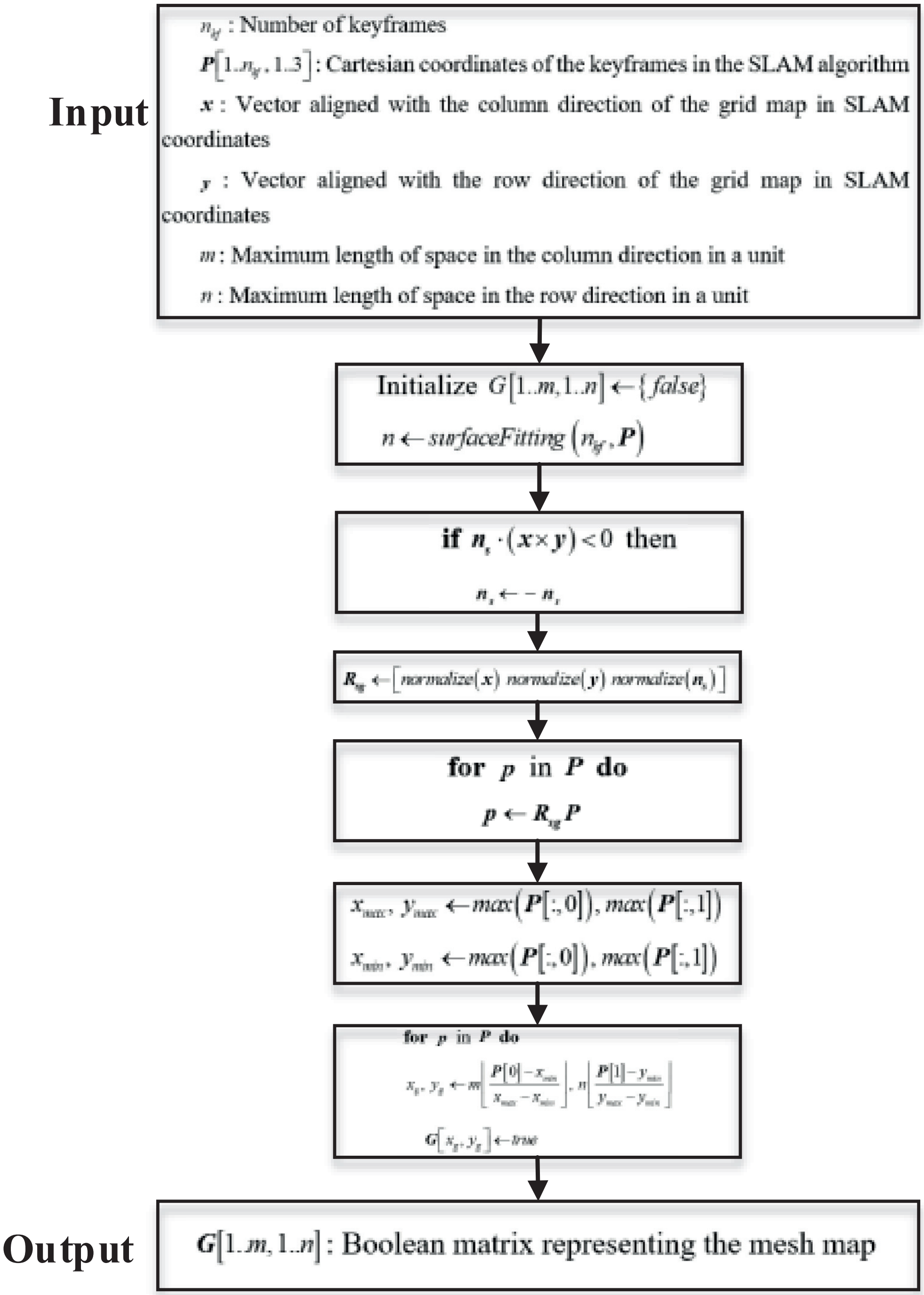
(3) There should be no less than one keyframe captured for an area accessible to robots.

The meshing procedure includes surface matching onto the keyframe coordinates, projecting the coordinates onto a mesh map, checking if each mesh has got at least one keyframe being projected onto, and setting the mesh value to true if yes. Details are described in Algorithm 1 with an example in Fig. 6 using monocular ORBSLAM2 [139]. A redundant intermediate output is a normal vector of the mapped plane, and it could be avoided by applying a mesh map that labels the accessibility of areas. Thus, this method provides convenience to route planning.

In this research, we reduce the feature data size by a filtering algorithm - Mesh-SLAM. Mesh-SLAM only keep the key features and corresponding mesh map. In this case, around 60% - 80% feature data is discarded depending on feature density in different frames. This Mesh-SLAM is designed for specific situation – large area with significantly similar or redundant features, like most agriculture scenarios. Finally, we are able to balance the trade-off between excessive feature data of a large farm and bandwidth constrain by either hardware or Wi-Fi Communications protocol.

2.3.3 Mesh Projection

The step of projecting SLAM coordinates to mesh coordinates involves projecting both the position and orientation as shown in Fig. 6.



Input

n_{kf} : Number of keyframes

$\boldsymbol{P}[1..n_{kf}, 1..3]$: Cartesian coordinates of the keyframes in the SLAM algorithm

$\boldsymbol{x}$: Vector aligned with the column direction of the grid map in SLAM coordinates

$\boldsymbol{y}$: Vector aligned with the row direction of the grid map in SLAM coordinates

m : Maximum length of space in the column direction in a unit

n : Maximum length of space in the row direction in a unit

Initialize $G[1..m, 1..n] \leftarrow \{false\}$

$n \leftarrow surfaceFitting(n_{kf}, \boldsymbol{P})$

if $\boldsymbol{n}_s \cdot (\boldsymbol{x} \times \boldsymbol{y}) < 0$ **then**

$\boldsymbol{n}_s \leftarrow -\boldsymbol{n}_s$

$\boldsymbol{R}_{sg} \leftarrow [normalize(\boldsymbol{x})\ normalize(\boldsymbol{y})\ normalize(\boldsymbol{n}_s)]$

for p in $\boldsymbol{P}$ **do**

$\boldsymbol{p} \leftarrow \boldsymbol{R}_{sg}\boldsymbol{P}$

$x_{max}, y_{max} \leftarrow max(\boldsymbol{P}[:,0]), max(\boldsymbol{P}[:,1])$

$x_{min}, y_{min} \leftarrow max(\boldsymbol{P}[:,0]), max(\boldsymbol{P}[:,1])$

for p in $\boldsymbol{P}$ **do**

$x_g, y_g \leftarrow m\left\lfloor \frac{\boldsymbol{P}[0]-x_{min}}{x_{max}-x_{min}} \right\rfloor, n\left\lfloor \frac{\boldsymbol{P}[1]-y_{min}}{y_{max}-y_{min}} \right\rfloor$

$\boldsymbol{G}[x_g, y_g] \leftarrow true$

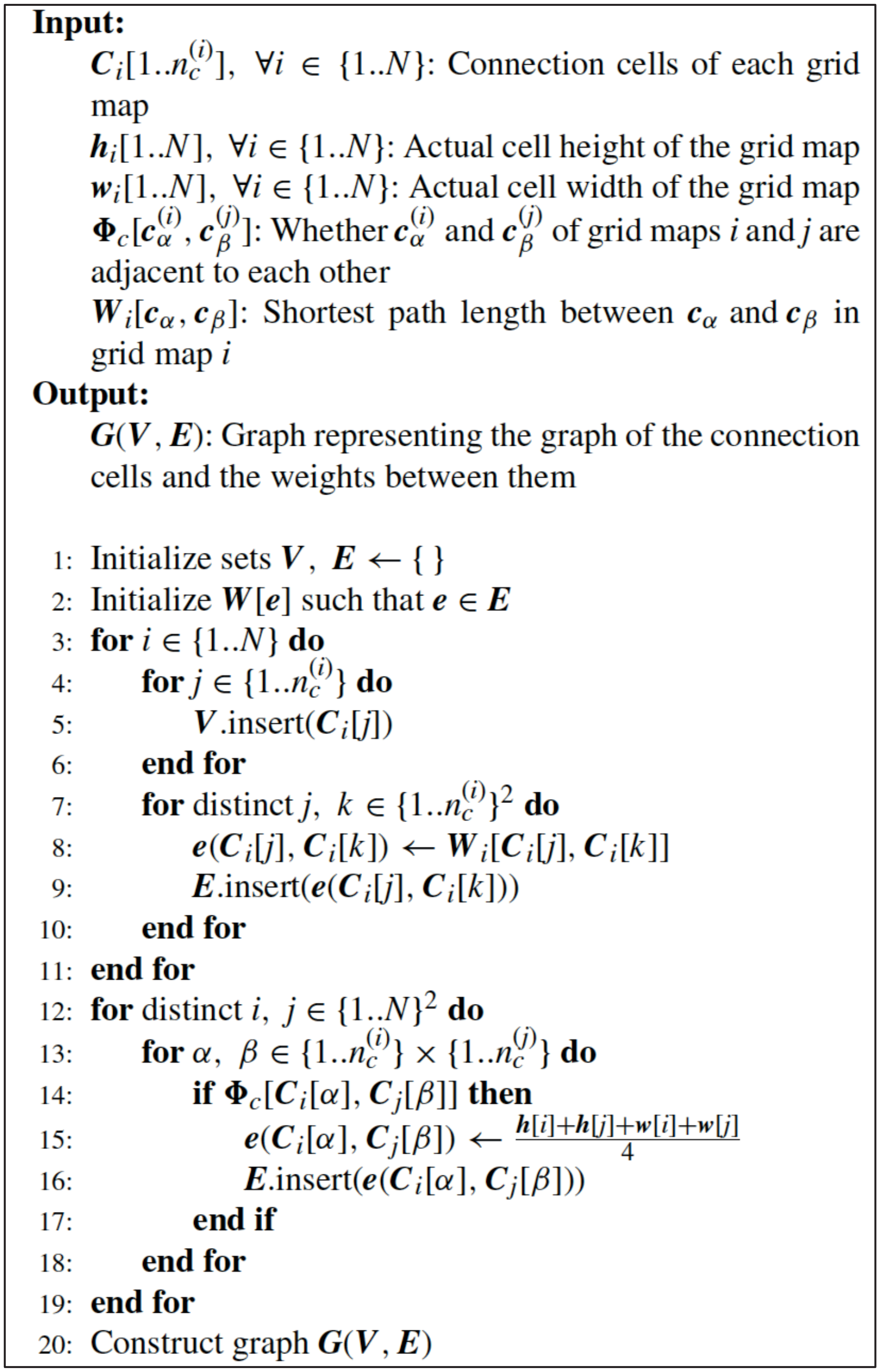
Output

$\boldsymbol{G}[1..m, 1..n]$: Boolean matrix representing the mesh map

Figure 6 mn-Scaled Meshing algorithm

The projection method for projecting position and keyframes onto the mesh coordinate is identical, shown in Fig. 6. The cell for a SLAM coordinate p is $\left(x_g, y_g\right)$.

Projecting SLAM orientation onto the mesh map is a necessary step to provide accurate navigation. The three-dimensional orientation is converted onto a planer map. Since each unit in the mesh is neighbored by 8 units, the neighboring units are numbered from 0 to 7. The forward vector of the mesh map is $v_g = R_{sg}v$ given v is the forward vector of the camera of SLAM, as described in Fig.6. Correspondingly, the orientation is calculated by the projection of the direction and coordinate. After we built a Mesh-map using the algorithm described in Fig.6, a boolean matrix of the viable cells/positions is created. Since the ratio between real-world scale and the size of each unit in the Mash-map is known, we developed a route planning algorithm to address the undirected weighted graph problem. Fig. 7 illustrates the details of the algorithm. As seen in the figure, a Dijkstra or Bellman-Ford algorithm are good choices for this shortest path route planning problem with OD (origin - destination).



Input:

$\boldsymbol{C}_i[1..n_c^{(i)}],\ \forall i \in \{1..N\}$: Connection cells of each grid map

$\boldsymbol{h}_i[1..N],\ \forall i \in \{1..N\}$: Actual cell height of the grid map

$\boldsymbol{w}_i[1..N],\ \forall i \in \{1..N\}$: Actual cell width of the grid map

$\boldsymbol{\Phi}_c[\boldsymbol{c}_\alpha^{(i)}, \boldsymbol{c}_\beta^{(j)}]$: Whether $\boldsymbol{c}_\alpha^{(i)}$ and $\boldsymbol{c}_\beta^{(j)}$ of grid maps i and j are adjacent to each other

$\boldsymbol{W}_i[\boldsymbol{c}_\alpha, \boldsymbol{c}_\beta]$: Shortest path length between $\boldsymbol{c}_\alpha$ and $\boldsymbol{c}_\beta$ in grid map i

Output:

$\boldsymbol{G}(\boldsymbol{V}, \boldsymbol{E})$: Graph representing the graph of the connection cells and the weights between them

```
Initialize sets $\boldsymbol{V},\ \boldsymbol{E} \leftarrow \{\ \}$
Initialize $\boldsymbol{W}[\boldsymbol{e}]$ such that $\boldsymbol{e} \in \boldsymbol{E}$
for $i \in \{1..N\}$ do
    for $j \in \{1..n_c^{(i)}\}$ do
        $\boldsymbol{V}$.insert($\boldsymbol{C}_i[j]$)
    end for
    for distinct $j,\ k \in \{1..n_c^{(i)}\}^2$ do
        $\boldsymbol{e}(\boldsymbol{C}_i[j], \boldsymbol{C}_i[k]) \leftarrow \boldsymbol{W}_i[\boldsymbol{C}_i[j], \boldsymbol{C}_i[k]]$
        $\boldsymbol{E}$.insert($\boldsymbol{e}(\boldsymbol{C}_i[j], \boldsymbol{C}_i[k])$)
    end for
end for
for distinct $i,\ j \in \{1..N\}^2$ do
    for $\alpha,\ \beta \in \{1..n_c^{(i)}\} \times \{1..n_c^{(j)}\}$ do
        if $\boldsymbol{\Phi}_c[\boldsymbol{C}_i[\alpha], \boldsymbol{C}_j[\beta]]$ then
            $\boldsymbol{e}(\boldsymbol{C}_i[\alpha], \boldsymbol{C}_j[\beta]) \leftarrow \frac{\boldsymbol{h}[i]+\boldsymbol{h}[j]+\boldsymbol{w}[i]+\boldsymbol{w}[j]}{4}$
            $\boldsymbol{E}$.insert($\boldsymbol{e}(\boldsymbol{C}_i[\alpha], \boldsymbol{C}_j[\beta])$)
        end if
    end for
end for
Construct graph $\boldsymbol{G}(\boldsymbol{V}, \boldsymbol{E})$
```

Figure 7 Route planning algorithm based on the Mesh-map

2.4 Experiment Design and Data Association

2.4.1 Test Location

The case study is conducted at the West Madison Agricultural Research Station (Madison, WI, USA). The Research Station is about 2.75 acres and there are 34 kinds of plants. Fig. 8 shows an aerial view of the station layout and Fig. 9 is an abstract map showing the testing areas. Fig. 10 gives a direct view for the real experiment scenario from the camera.

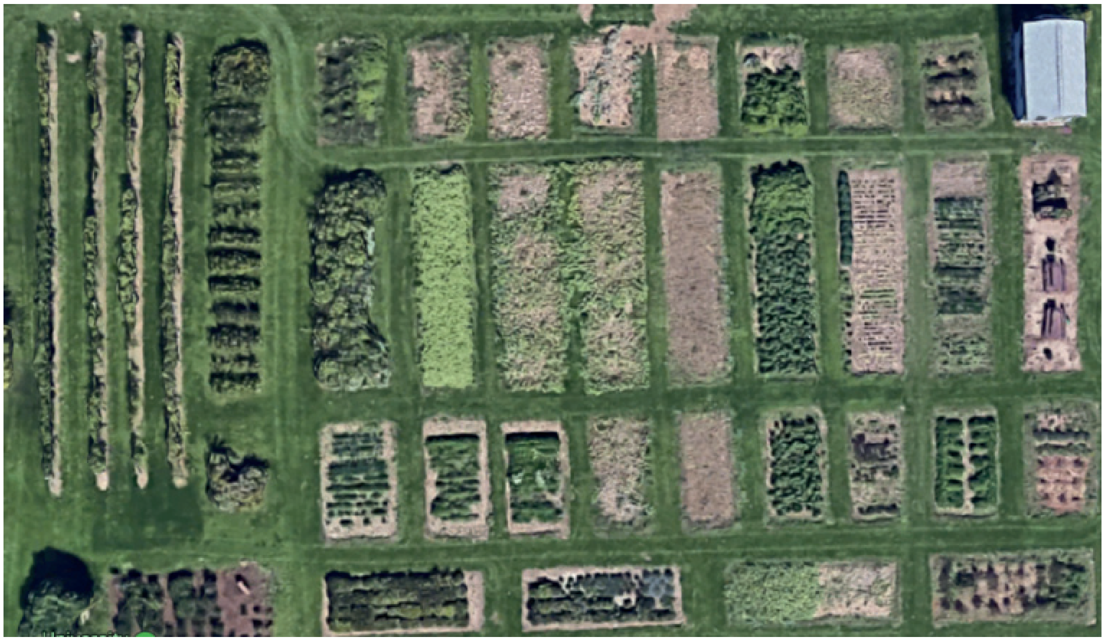


Figure 8 Map of the testing area.

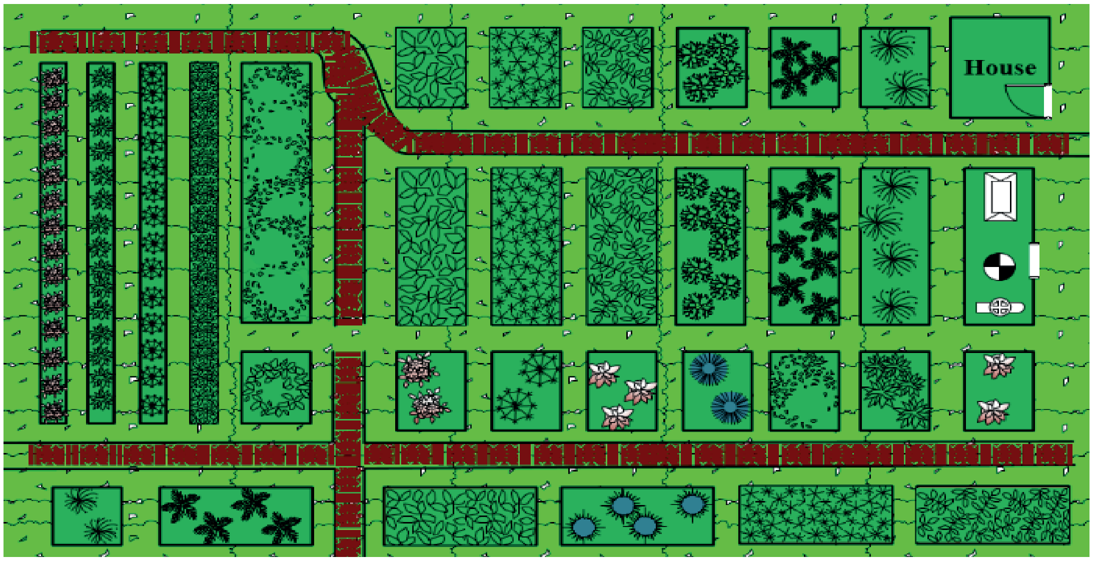


Figure 9 Abstract map of the testing area

2.4.2 Hardware Configuration

Camera: In this work, we choose an RGB camera as the monocular camera to collect data from the farm and conduct various experiments. More specifically, we choose the Logitech-C922x-Pro USB camera in this work with frame rate of 15fps. Logitech C-series USB cameras are widely chosen as monocular cameras when building prototype systems for SLAM and ROS [138], [139]. Also, the H.264 encoder in this camera offers high resolution and frame rate with a reasonable price compared to general RGB cameras.

Besides, we decided to use monocular cameras for the proposed system based on the following reasons. First, a monocular camera is cheaper than other types of cameras like stereo and RGB-D cameras. As the system is mainly used in outdoor environments, it is possible that the system needs to work under extreme weather conditions. So, we can easily replace any broken parts for the system with a lower cost. Second, stereo and RGB-D usually require more computing and energy resources to process collected data. Farming robots have limited computing resources and battery capacities. Moreover, since this camera is used in an outdoor environment which limits the RGB-D camera because the lighting condition is not structured light. Last, machine vision cameras and CCD cameras are overwhelming in terms of cost and performance for this system, since our robot working scope is for mapping and navigation. Hence, it is important to use energy efficient hardware and develop energy and computational efficient software. As discussed in our previous work [140], current monocular-based SLAM systems are not suitable to run on portable devices for our purpose.

Edge node and Server:

Table 2 System implementation hardware configuration

IoT	Item	Description	Item	Description
Cloud Server	CPU	Intel Core i7 8700K	Bandwidth	~500Mbps
	GPU	NVIDIA RTX 1080ti	Network	Wi-Fi 802.11ac, UDP
	RAM	64GB	System	Ubuntu16.04，x86_64
Edge Node	CPU	Hexcore ARMv8 64-bit CPU	Bandwidth	~500Mbps
	GPU	256-core Pascal GPU	Network	Wi-Fi 802.11ac, UDP
	RAM	8GB	System	Ubuntu16.04，x86_64

In terms of computing hardware of this IoT system, different hardware configurations were chosen for Cloud server and Edge nodes as shown in Table 2. Since we focus on leveraging the advantages of edge computing platforms to design a system that can work in a large area. The Cloud server is sufficient in terms of GPU and RAM in this test scenario. Meanwhile, the bandwidth and CPU could be the restriction if the system need to scale up. Taking advantage of this system, simply upgrading the hardware will solve it.

2.4.3 Data Collection

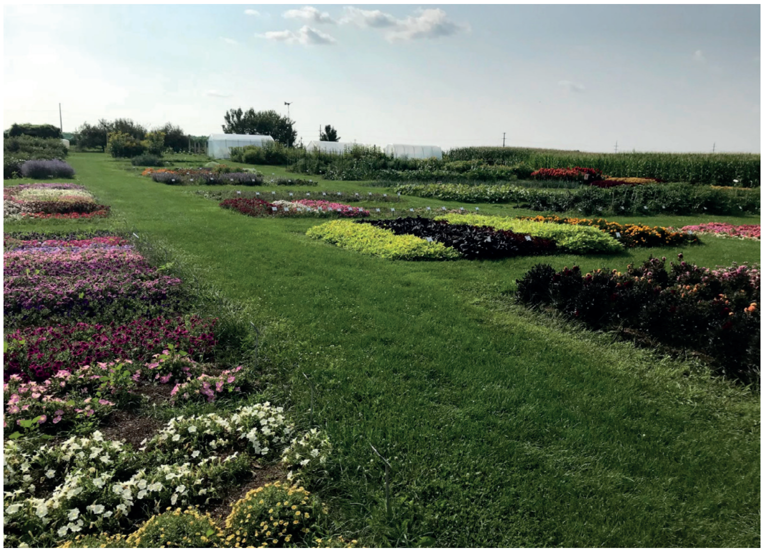


Figure 10 Experiment farm real view

We collected the data across all seasons because one of the research questions is how the plants' shape change may affect the mapping and localization results. The frequency of the experiments is performed in accordance to the plants’ growth rate. The sample collection date is shown in Table 3.

We summarized the total number of data capture and time duration for each capture in the following table. It took around 6.5 hours creating the map for the first time and then the time needed for updating maps became less and less. As the farm changes, the system only needs to update corresponding features or missing features when necessary, so it needs less time than creating the map. In other words, the system doesn’t need a significant amount of time to maintain the map.

Table 3 data collection time and context

Sample	Date Measured	Time cost(hour)	Context
1	4/5/2018	6-7	Create the map
2	15/5/2018	4-5	Update the map
3	2/6/2018	3-4	Update the map
4	15/6/2018	2-3	Update the map
5	8/7/2018	2-3	Update the map

2.4.4 System Design

The mapping system is designed with three components including: mobile robots, edge nodes, and a cloud server. In Fig. 3, a cloud server controls the edge nodes, and the presented area is managed by four edge nodes. The mobile robot is controlled by the edge node.

In our work, the cloud server is running on a machine located in our lab. And our cloud services can be easily migrated to other cloud servers, e.g., AWS, Digital Ocean, and so on. For the edge node, we use an existing framework, the edge-box is designed based on the Docker container technology, which means it can be run on any device that supports Docker. Hence, we can update and upgrade edge nodes whenever needed. Lastly, we use a commercial-off-the-shelf USB camera for video collection on mobile robots, it is a plug-and-play design. There are no specific requirements on the robot, so any robot with a good mileage and power supply can be the robot in our system. In terms of system software, there are three major modules, including data collection, data transmission, and data analysis. Different modules are responsible for different tasks, each module can be changed or updated without interrupting other modules. Additionally, each module can be updated over the air as long as a suitable internet connection is maintained.

Robot vehicles: Only two tasks are designed for robot vehicles and no complex computation is involved. One of the tasks is to transfer the message of destination and its surrounding scenarios captured by its camera to the “region-manager” edge node. The other task is to get the feedback from the edge node with guidance to the destination.

We use a commercial-off-the-shelf USB camera for video collection on robot vehicles, it is a plug-and-play design. There are no specific requirements on the robot vehicle, so any form of mobile vehicle with a good mileage can be the robot vehicle in our system.

The robot needs the map of the farm for the first time building the SLAM mesh map. After that, it uses a controllable itinerary plan to update the map when necessary. The purpose of having an itinerary plan to make sure all the paths in the farm have been covered. If the system collects enough features to build the SLAM mesh map, then the process is finished. Otherwise, the system will find out which points on the map don't have enough features, and design an itinerary plan to collect data until we have enough features to build the mesh map.

Edge nodes: The procedure of processing a navigation task by edge nodes is shown in Fig. 4. Once the edge node is activated, a mobile robot sends a navigation request and a sequence of frames with surrounding environment to the connected edge node. Each image is processed by SLAM to calibrate the coordinate and projected onto a mesh map. The mesh location is utilized for navigation. A cloud server is required if the destination is not in the “managing region” of the connected edge node. The last step was to send back the planned route to robots.

Cloud server: The cloud server provides two services: global navigation and map maintenance. The global navigation task was synchronized with the edge node request, but the map maintenance has to be asynchronous because the growing status change of the plants was involved which requires a high computation tracking algorithm and appropriate hardware to process such tasks (e.g., GPU).

Existing cloud-based applications execute computation tasks on the centralized cloud servers. All the data need to be uploaded to the cloud for further analysis. With the increasing number of IoT devices, a large volume of data is produced every second and it is hard to upload and process such data on the cloud server. Edge computing is a newly proposed concept that computation is done at the edge of the network, close to where the data is being generated. Raw data can be processed locally and only update necessary data to the cloud. Hence, network bandwidth can be saved and applications can have a better response time. To sum up, we focus on the scalability and flexibility when designing the system from both hardware and software aspects. We believe our proposed design make it easy to adapt most future needs.

2.4.5 System Implementation

A SLAM implementation, ORB-SLAM2 [141], was chosen to build this system due to its advantage of engineering convenience. ORB-SLAM2 uses the Boost Serialization library [142] to save and read maps. Other monocular SLAM implementations are also applicable in this system if properly adjusted. The details of the measurement for each section are described in the corresponding paragraphs.

Our system costs depend on the size of the farmland and the service/application scenarios. Since this system is easily scalable, people can calculate cost according to this brief explanation of the cost for each component.

Throughput: The bandwidth between the local area network (LAN) and the cloud server is likely the bottleneck for the number of clients to scale up. Hence the traditional client-cloud server paradigm is not suitable for our purpose. For example Zhu et al. [143] claimed that Network heterogeneity: As different networks, such as Internet, wireless local area network (LAN), and third generation wireless network, have different network characteristics, such as bandwidth, delay, and jitter, the cloud shall adapt multimedia contents for optimal delivery to various types of devices with different network bandwidths and latencies. As a result, we proposed the client-edge-cloud architecture in this work. In our design, each robot is configured to stream captured images (640*480) at a frame rate of 6. The robot first streams images to the edge and then to the cloud. With the help of the edge, we can achieve a higher throughput than traditional setups as images can be preprocessed at the edge, which could significantly reduce the size of data that is needed to be uploaded to the cloud.

Load Testing: We have conducted an experiment, with smartphones mimicking the robot streaming, to study the system performances under different loads. As shown in Fig. 12, we have studied how the system performs when there are multiple edge nodes and different numbers of users. Our evaluation results show that the more working robots (more than 16), the lower CPU usage, and the CPU usage of the centralized server setting decreases more rapidly. The CPU usages of both edge node settings increase at first and decrease when there are more than 26 working robots. Hence, we could achieve a good performance of 2 edge nodes. If the system needs to support more robots, we could add more edge nodes to the system.

Reliability: As mentioned previously, the map needs to be updated as the crop grows or farm changes. Updating the map periodically can significantly improve the reliability and robustness of the system. We draw a figure to illustrates how to maintain the map in Fig. 11.

Eventually, we design an automatic map maintenance mechanism as demonstrated in Fig. 11. As the mobile robot streams images to the edge node, if the SLAM localization is successful, the edge node buffers the most recent images in a short period. When a localization failure occurs, the edge node keeps collecting images for some time, and send these images along with the buffered ones to the cloud server. Although the images with other dynamic objects can be leveraged to perform future analysis, we abandon them as well, since we are interested in the environment texture. The image sequence without dynamic objects is sent to SLAM with mapping mode activated. The cloud server has copies of all the SLAM maps on the edge devices and can directly load them to SLAM for map maintenance. The SLAM system handles the images with the normal routine. If the map is successfully updated, it will be pushed to edge nodes.

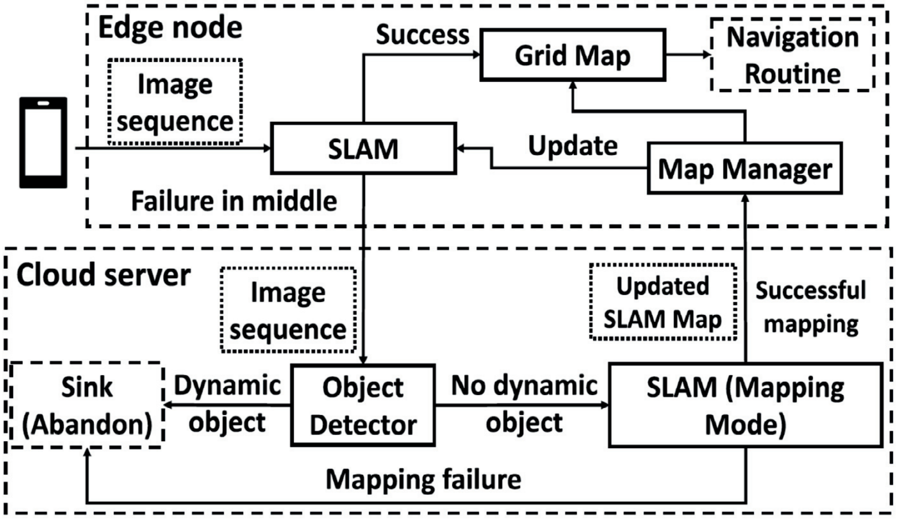


Figure 11 The flowchart of map maintenance

2.5 Results and Discussion

2.5.1 Accuracy

The accuracy performance of the mapping between SLAM and the Mesh is an important criterion in this system. This accuracy is evaluated under various scenarios and mesh densities.

Experiment Setup The testing field is described in the experiment section. During the process of calibrating the SLAM map onto the mesh map, the cameras were set to the space boundary at which location keyframes were captured. And this step ensures the edges of the space, the dimension, and the direction of the mesh map are consistent.

Table 2 Results from the accuracy experiment

Cell length (approximate) (cm)	30	60
Localization success frequency (%)	84.7	89.3
RMSE (cm)	19.5	0
Maximal error (cm)	36.9	0
Orientation accuracy (cm)	100	100

Note: The GPS result is not listed here because the GPS is not well functional within 60 cm. Orientation includes 8 directions separated by 45 degrees.

This effort ensures an accurate calibration and precise output. The ORBSLAM2 has a farm accuracy below 5cm in the monocular mode [144]. The validity of a location within a mesh relies on if a keyframe is captured in that mesh unit. Therefore, the mesh unit size should not be too small to guarantee a keyframe is associated especially when the density of keyframes is unknown. A small-scale preliminary experiment was designed with the mesh unit dimension set to be 30 cm, which is empirically safe for this application. The final experiment sets the mesh unit with 30 cm and 60 cm respectively as shown in Fig. 12.

The mesh is perfectly square due to the aspect ratio of the ground, so the actual dimension is considered. Both experiments (30 cm and 60 cm) are conducted with the same route. An edge node is set up and connected with the robot. The OpenCV camera calibration model [145], [146] is used to set up the conversion between the robot and the camera. Each experiment was repeated three times, and in each execution, the robot moves towards the next grid along the path (red line shown in Fig. 12), and the edge node computes the mesh coordinates and the ground truth. Fig. 13 shows the real-time path merging results when building the map. Correspondingly, these merged paths could possibly be assigned into different grids when we scale cell size. Additional real-time mapping construction screen shot images are in Appendix 2.

Result Table 4 shows the results under both settings. The accuracy is calculated based on the localization success frequency. In both experiments, a same path is selected. And we marked this path on the grass as the ground truth for future loss calculation. When we finished building the map as a closed loop. We repeat this process with the same path for several times until the map is built. It is the ratio of the recordings of a correct localization. Another measurement is the accurate localization frequency, which is the proportion of the localization records over the correct localization records. The difference between the computed mesh coordinates and the ground truth is calculated by the Root-Mean-Square (RMSE) metric, which is shown below. Accurate localization is defined if the error is acceptable depending on plant size and farm scale.

$$\text{RMSE}=\sqrt{\frac{\sum_{i=1}^{m}\left(\left(h\left(x_i - x_i^{(g)}\right)\right)^2+\left(w\left(y_i - y_i^{(g)}\right)\right)^2\right)}{m}} \tag{1}$$

Here, the width and height of a unit of mesh are w and h. The coordinates of the server computation result are (x_i, y_i), while $(x_i^{(g)}, y_i^{(g)})$ shows the ground truth. m is the quantity of successful localization records. A maximal error happens if it is measured from the center of the localization output to the actual location using Euclidean distance.

The table shows 84.7% correct localization for the 30 cm group, and with 89.3% correct for the 60 cm group. The RMSA is 19.5 cm and 0 cm for 30 cm and 60 cm group respectively. The maximum error is 36.9 cm in the 30 cm group and 0 for the 60 cm group. Because there are unflat surfaces, so individual frame localizatio error can be as large as 36.9, which is in the wrong grid. However, this kind of errors happen less than 20% as shown in Table 4. During passing by each grid, several frames are captured and sent to be processed, which means the final localization is still accuracy with each grid as a unit as long as most frames are accurately locoalized. The calculated orientation has been verified that they both are constant with the ground truth.

Analysis This experiment was carried out with two goals: to show the accuracy of the algorithm, and to understand the sensitivity of the Mesh map parameters. The results show the localization success frequencies of both groups are similar. Hence, SLAM is the only key to decide the success of localization. A unit in the mash position could always be calculated if the given SLAM could provide the SLAM map coordinates. Thus, localization failures are mostly caused by SLAM, which is further discussed in the next section.

In terms of accuracy, the 60 cm group shows the results are matching with the ground truth better. Even with the 30 cm dimension, the maximum error is located at the neighbor of the actual value. There are two possible reasons for the error: SLAM localization failures, and errors of mapping projection between SLAM and mesh. In the aspect of direction, the results are 100% correct for both groups in the 8-direction system. Overall, the experiment shows the algorithm obtains a high accuracy in localization with a 60 cm unit and high performance with 8 directions.

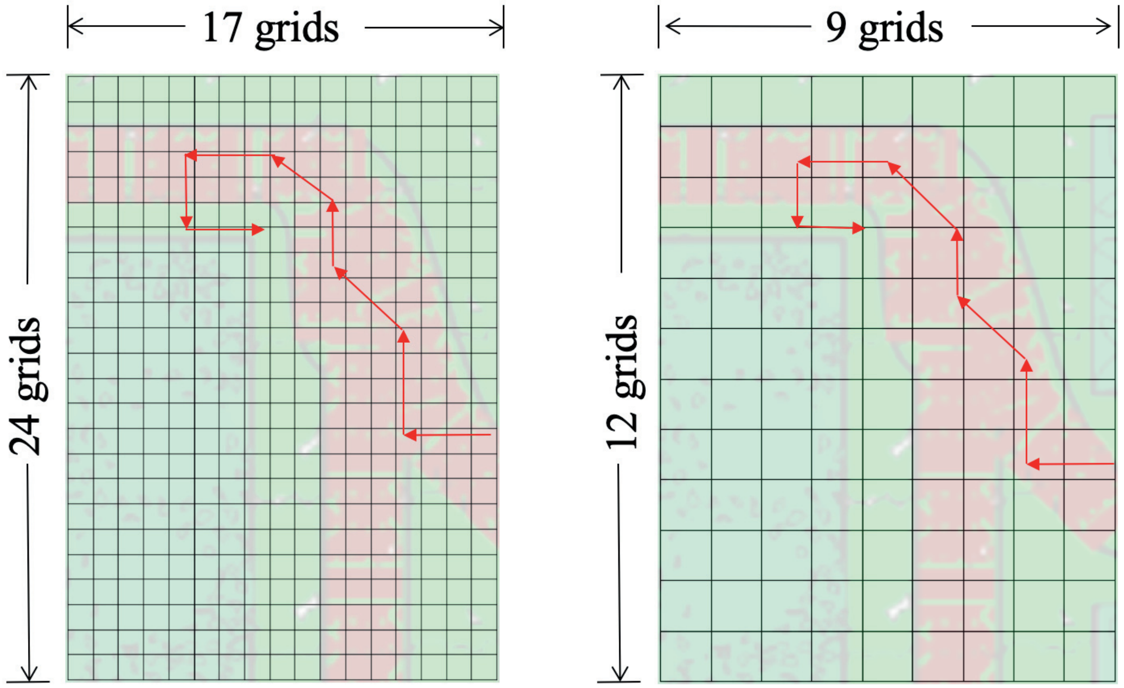


Figure 12 The configuration of the accuracy experiment with a unit of 30cm-cells (left) and 60cm-cells (right)

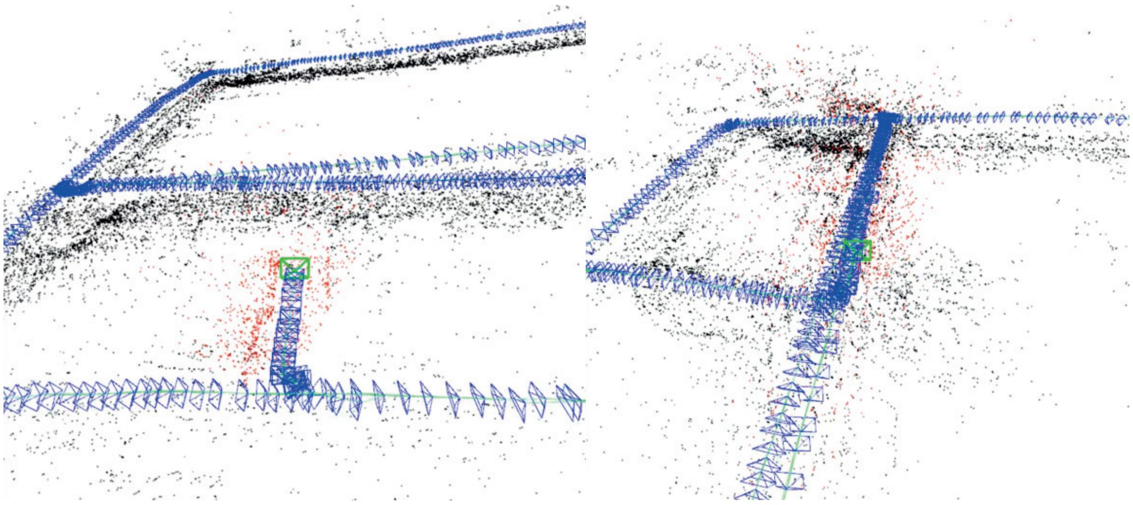


Figure 13 Building the SLAM map including (left) the map before path merging and (right) after path merging

2.5.1 IoT - Scalability and Feasibility for Farm

The capacity of the edge computing framework was evaluated by testing the time gap between neighboring responses on the robot side when the volume of the simultaneous requests from the robots is enlarged step-by-step. This experiment was constructed on a centralized cloud server with one or two edge nodes. The results are shown in Fig.14.

The cumulative distribution function of time intervals between successive responses from the server and the edge node(s) with different numbers of concurrent users is shown in Fig. 14.

These time intervals can be treated as user waiting time. To calculate the user waiting time, each response's timestamp is subtracted by the previous one. When there are two working edge nodes, the configuration yields much smaller waiting time than other settings. In general, the system can gain more advantages when more edge nodes are available. What's more, the waiting time is smaller than that under the centralized settings when only a single edge node is available. With the number of users increasing from 26 to 36, the centralized service has a significant deterioration. For about 14.8% of cases, a robot needs to wait for at least 2 seconds and even needs to wait for more than 3 seconds for around 5% of cases. If there is one edge node available, 2-second waiting time appears in 8.7% of cases and 1.6% for ~3-second waiting time. Less than 0.5% of cases experienced a more than 2 second waiting time when there are two edge nodes available. Fig. 12 summarizes the minimal, maximal and average CPU usages. CPU usages for 6-robot group is about 193.7%, 201.2% and 124.3% for the single, double edge node and central server respectively. When there are 16 robots (we used multiple smart phone to mimic more robots scenarios), the CPU usages reach to the highest (266.5% and 263.8%) for both edge and centralized settings. The more working robots (more than 16), the lower CPU usage, and the CPU usage of the centralized server setting decreases more rapidly. The CPU usages of both edge node settings increase at first and decrease when there are more than 26 working robots.

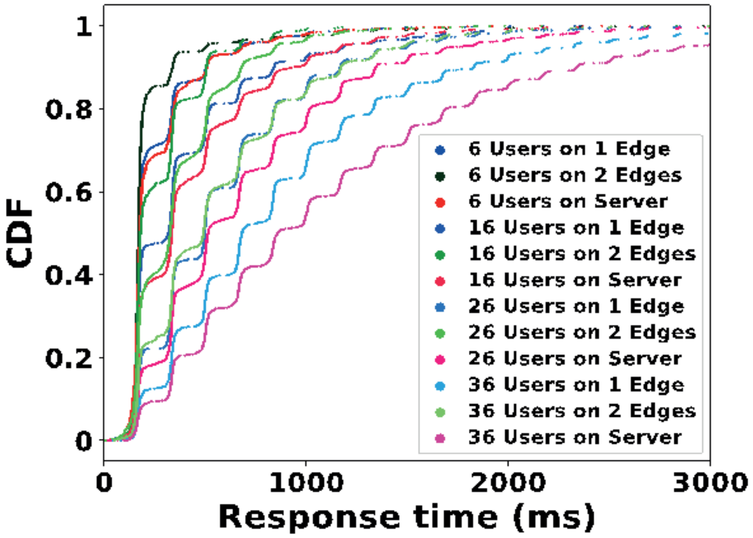


Figure 14 The CDF of the time intervals between responses.   Note: User means a working robot

Results Here, a user means a working robot. The time gap can be treated as the user waiting time as well. It is the time interval between the timestamps of each neighboring response. For each experimented user volume, the user waiting time produced by two nodes is much smaller than that of other settings. It suggests that the more the nodes were used, the better the performance, with the time gap of using one node smaller than that with the centralized cloud setting. When the quantity of the robots grows from 10 to 26, the centralized service setting demonstrates a significant performance decrease, wherein 14.8% of cases a robot would have to wait for no less than two seconds, and the cases to stand by for longer than three seconds is 5.0%. For comparison, chances are 8.7% and 1.6% for one-node setting, and the probability is less than 0.5% for the two-node setting. The CPU usage distribution concerning the number of robots is shown in Fig.15.

Because the CPU has multiple cores, so the CPU usage can be large than 100%. With 6 robots, the CPU usage is 193.7% for the one-node setting, 201.2% for the two-node setting, and 124.3% for the central cloud server setting. And it is 266.5% and 263.8% for the one-node and centralized cloud server setting with 16 robots. The CPU usage of the one-node and centralized cloud server setting decreases as the robots' number increases.

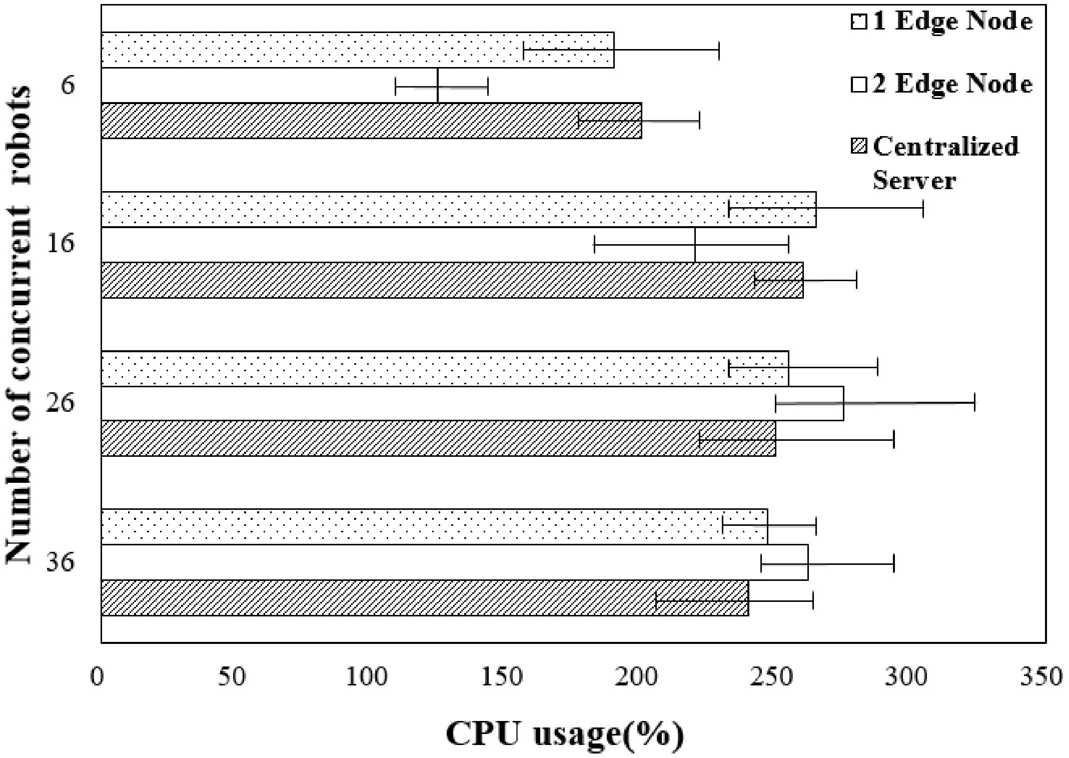


Figure 15 CPU usages in each experiment configuration

Analysis A longer robot waiting time corresponds to worse performance of remote control and processing time. The experiment shows a user would have a higher chance to wait for more than 5 seconds to get the following response if the volume of concurrent working robots increases. There are two possible reasons for this downgraded performance: 1) low computation power, 2) missing packets in the network. However, the experiment shows that the CPU usage decreases when the robot volume increases to 16 and above, which is contradictory. This suggests the longer waiting time is caused by missing packets that deliver images and requests when the robot volume is large. This also explains why the centralized cloud server setting performs worse than edge settings. All users send out UDP packets at about the same frequency. So, if the number of concurrent users becomes larger, the possibility that a communication backlog happens becomes higher. This results in more requests and less responses, and thus longer user waiting time. The poor performance of the centralized cloud server setting is because the centralized setting is designed with an extra hop in the connection between the Wi-Fi router and the user, leading to a higher probability of missing packets and longer traveling distances.

The experiment also reveals that under a two-node setting, the system service area could be enlarged under the same request-response criteria. This conclusion relies on the premise that the number of robots connected to each node is equal, as the performance decreases if the distribution is not even. However, even in the worst-case scenario, where all robots are connected to a single node, the performance of the one-node setting is still higher than that of a centralized server setting, though only by a small amount. Thus, we conclude that the edge computing scheme proposed in this paper is capable of providing a better concurrency than the traditional centralized cloud server setting.

2.6 Conclusions

In this chapter, a mapping algorithm and a vision-based farm navigation scheme have been addressed. A cloud-assisted architecture was utilized to disperse the computation load and network communication between multiple edge nodes and a single cloud server. Additionally, a mesh map was presented which avoids the prior information of the testing land. The experiment shows 1) The maximum of localization error is 60 cm, which is among the top performance with other systems 2) This scheme allows larger capacity than the centralized server setting 3) The map could be more frequently updated with different scenarios by taking advantage of this IoT architecture's clever network distribution.

Chapter 3 Sensing: Augmenting Map Insight based on Advanced Sensors Data Fusion

This chapter has been submitted to journal Sustainability entitled "Terrain Analytics for Precision Agriculture with Automated Vehicle Sensors and Data Fusion".

3.1 Introduction

Precision agriculture includes precise irrigation quantity, correct and appropriate application of chemicals, weed cleaning, where topological characteristics of the crop field is a key component needs to be considered. For instance, quantity of irrigation varies drastically due to the change of terrain slops. Hussnain et al. [35] pointed out that low areas of crop field are likely to collect more water from either the irrigation and rainfall, meaning that irrigation for those areas are better to be less compared to other areas with high slopes. Mareeles et al. [36] concluded that precision agriculture in terms of irrigation system relies heavily on the topological characteristics of the crop field.

Topological terrain attributes also influence the efficiency of implementing large agriculture machines. Machine and tractor fleet used in agriculture improve the efficiency of farming process, while they also require appropriate terrain attributes to be able to perform and maneuver (Myalo et al., [147]). Without much information about the specific topological characteristics of the crop field needed to use large farming tractors could make the machines stuck in the field and reduce the farming speed, resulting in huge loss for farmers. Therefore, information collection for detail topological characteristics is essential in precision agriculture.

Adverse topological characteristics detection in crop field for the precision agriculture is another key factor. Large machine for crop watering, applying pesticide to control diseases, and harvesting are commonly used for scaling crop production. When the terrain is not even, large machines need a way to know if they can be applied smoothly. Otherwise, the stuck machines in the crop fields could cause serious economic loss either from the machines unable to maneuver shortly or delay the crop production process and hence prevent the implementing of precision agriculture.

Advanced sensors and computer vision techniques provide an opportunity to collect topological characteristics from the environment required by precision agriculture. Sensors are becoming smaller and more powerful (Nandurkar et al., [56]). In addition, the costs of sensors are decreased, enabling the widespread deployment in practice. The development of computer vision technique reaches the level that users can efficiently process collected data for decision making.

However, collecting all information from the crop field not only increase the burden of memory needed by the equipment, but also add more useless computing tasks. A method to extract critical and useful information from the field is needed. We propose a method that combines the advanced sensors and IMU as well as the algorithm to monitor and crop field in real time at a low cost and in a practical manner. In addition, the automated monitoring system also performs at field analysis of the relationship between the topological characteristics and crop growth to provide useful information for farmers to adjust their crop management strategies accordingly and in a timely manner.

In addition to the 3D map that we can build, additional information like the landform and more accurate topography is still requisite. The terrain will not only affect the growth of plants but also the management processing by robot farmers. For example, it significantly increases the processing time and fuel cost for a large machine if there is a lot of turning, and the unbalanced landform could result in different soil and water conservation. Also, the algorithms for map maintenance [148] are susceptible to adverse surface conditions, like unexpected pits, washouts from precipitation, steep slopes, or barriers. These drawbacks can limit the ability of our autonomous algorithms and further lead to the failure of localization. As a result, high-quality map building is challenged when attempting to include all these details [149]. Moreover, different landforms may result in various growing status, [150] for example, steep slopes are exposed to further and more severe erosion processes, such as gullies or mass movements.

Remote sensing plays a critical role in precision agriculture by allowing farmers to collect various types of information to help improve the quality and yields of crops. Optical remote sensing collects images with both visible and near infrared sensors. This sensing technology has been utilized in many aspects of crop production. Frolking et al. [151] developed new maps by combining the optical remote sensing and ground census data to investigate the diversity of rice production in China. Hall et al. [152] reviewed the applications of optical remoting sensing in viticulture by taking advantage of the information, such as soil structure and vine shape and size, from the collected images to ensure the quality and yields.

3.2 Methodology

3.2.1 IMU Sensors Data Collection Approach

Two IMU sensors, a gyroscope, and an accelerometer were used to measure the attitude of a robot vehicle [153]. Inertial sensors come with intrinsic noises. To improve their usability and accuracy, we designed a coordinate alignment algorithm on both slope surface and flat surface to detect sensor orientation changes and model stability. The proposed slope-aware algorithm first conduct coordinate alignment and estimate linear acceleration via dynamically withdrawing the gravity effect on recorded accelerometer readings. Then, it uses a clustering technique to identify relative orientation changes.

A gyroscope is an inertial sensor for measuring orientation based on the principles of angular momentum. However, because of noise jamming, temperature variation, and unstable force moment, algorithm drift error will occur and increase with time. Therefore, a gyroscope cannot be reliably used for a long time. A different option is to use an accelerometer is a device that measures proper acceleration. When the accelerometer is motionless, the attitude angles can be calculated based on the acceleration of gravity component in every axis via trigonometric functions.

Slope-Aware Alignment. The accuracy of coordinate alignment is mainly affected by the slope of the field. Hence, we develop a slope-aware coordinate alignment method to reduce or eliminate the negative effects caused by the slope. Traditional approaches fail to consider the slope, because they assume that all the motion data is through the origin motion data point, and calculate the fit curve based on all motion data [115]. However, these approaches could encounter chaos with a random rough surface. Therefore, our slope-aware approach dynamically segments the whole path into pieces and use each piece of the path as an independent input. Due to forces created by slopes, readings from each path deviate from the origin point. If we combine all the paths, we can estimate the slope and further improve the alignment accuracy. A rotation matrix will be derived by combining sensor readings from all paths. To derive the rotation matrix, we fitted the curve toe find the direction unit vector. Different from traditional approaches, we trained the horizontal unit vector for each segment and combined them by assigning different weights for each segment. One segment will be selected if the recorded data indicates the car is in motion. The more data points we can include in this segment, we can likely increase the statistical power of the measurement.

As shown in Fig. 16, the vector$[V_1, V_2, V_3]$represents the orientation of a gyroscope sensor and $\left[V_1', V_2', V_3'\right]$represents the orientation of a robot vehicle. The rotation matrix$R = \left[R_i, R_j, R_k\right]$can be estimated during the coordinate alignment process. In the rotation matrix R, R_i, R_j and R_k are the unit coordinate conversion vectors along each demission, such that $\left[V_1', V_2', V_3'\right] = [V_1, V_2, V_3] \times \left[R_i, R_j, R_k\right]$.

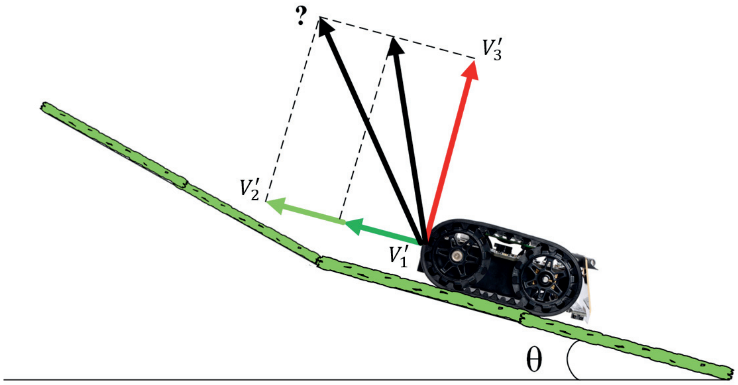


Figure 16 Robot motion detection

Different land surfaces can lead to various accelerations and further affect the moving speed and motion patterns. Hence, recognizing motion patterns can help us understand the running environment of the robot. In this work, we use the Dynamic Time Warping (DTW) algorithm to identify similar robot's motions with varying speeds and further detecting running environments. The DTW algorithm is well known for evaluating the similarity between two temporal sequences. To measure the similarity, temporal sequences are "warped" by shrinking or stretching in the time domain. To achieve the best performance, the training set should be carefully prepared and include representatives of different types of events as much as possible.

In order to improve detection accuracy, we choose to loosen some of the constraints of the DTW matching algorithms during the training process and also when conduction evaluations. As a result, our DTW algorithm can identify most of the motions, however, there will be some false positives. We think it is ok to have false positives as we can use other techniques to evaluate detected motions and further eliminate false positives, we leave this as future work. There are five categories of motion patterns in our research as are shown from Fig. 17-21.

(1) Flat

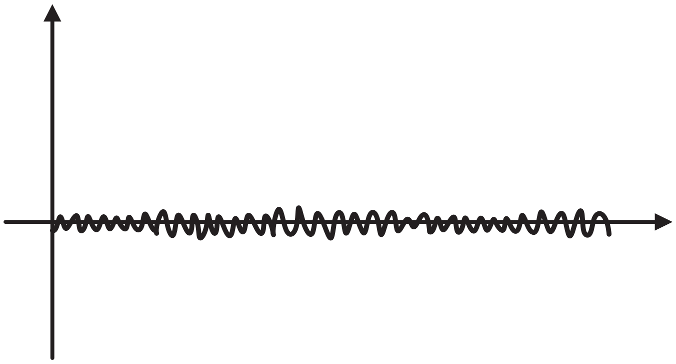


Figure 17 Motion sensor pattern moving on flat land

(2) Single slope

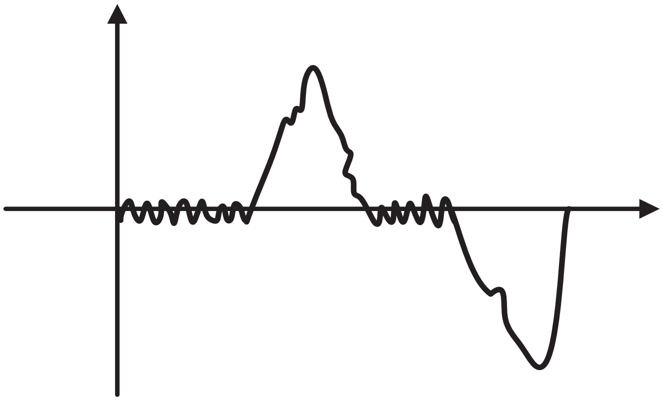


Figure 18 Motion sensor pattern moving over a slope

(3) Continuous hills

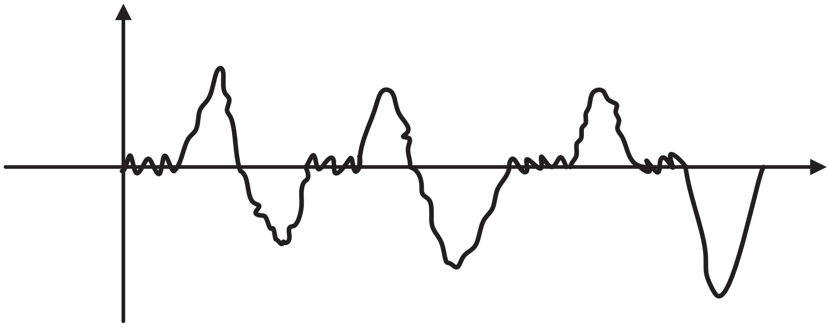


Figure 19 Motion sensor pattern moving over continuous hills

(4) Depression or Soil Erosion

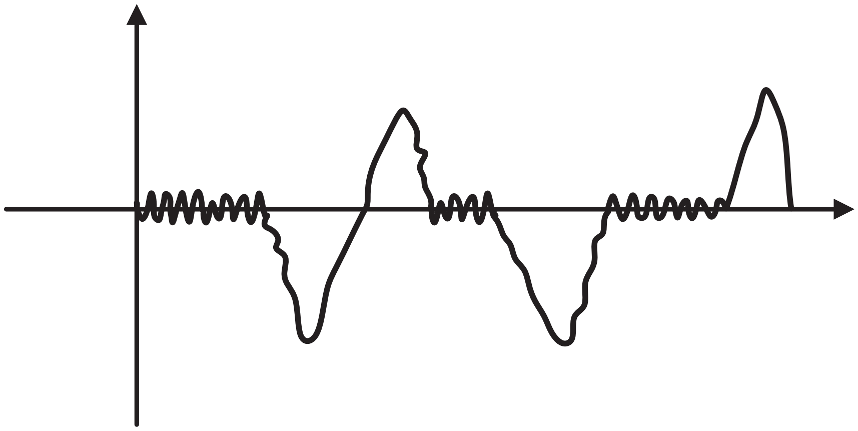


Figure 20 Motion sensor pattern moving across a depression

(5) Muddy

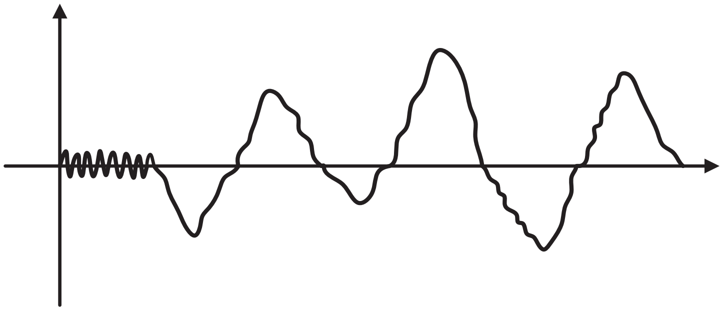


Figure 21 Motion sensor pattern moving on muddy land

3.2.2 Data Fusion Approach

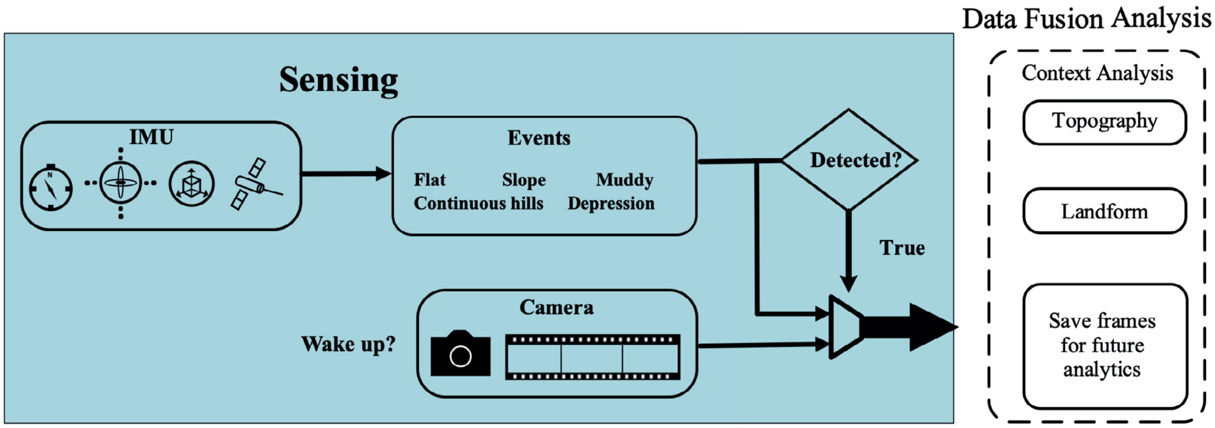


Figure 22 Architecture of multiple-sensor based sensing algorithm

The sensor fusion method avoids measurement limitations by using only a camera. An advanced data fusion method was used to integrate data from the camera, accelerometer, and gyroscope. In this study, two methods were discussed and compared. One is a self-adaptive complimentary PCA and the other one is the DTW.

Data collected from different sensors need to be synchronized as the clock on each device is different. After clock synchronization, we use Principal Component Analysis (PCA) to speed up the data analysis process and detect various activities promptly. PCA can extract the most important features from the collected dataset and convert high dimensional dataset to lower dimensions. PCA simplifies the dataset by discarding the least important features while maintaining the interpretability of variables.

Additionally, PCA also combines original variables in a way that only the most valuable features are retained. Hence, we extract features from our accelerometer & gyroscope dataset using the PCA algorithm. The algorithm is described as follows.

(1) Data Normalization

$$X^* = \frac{X - \mu}{\delta} \tag{2}$$

Where μ is the mean and σ is the standard deviation of all data.

(2) Covariance Matrix Calculation

$$Matrix(Covariance) = \begin{bmatrix} Var[X_1] & Cov[X_1, X_2] \\ Cov[X_2, X_1] & Var[X_2] \end{bmatrix} \tag{3}$$

Where X_1 is the accelerometer readings and X_2 is the gyroscope reading.

Note that $Var[X_1] = Cov[X_1, X_1]$ and $Var[X_2] = Cov[X_2, X_2]$.

$Matrix(Covariance)$ is the covariance matrix, which is a $d \times d$ matrix. The covariance matrix stores the covariance between two features. Once the covariance is established, PCA will perform the eigen decomposition on it. The covariance matrix can be calculated using equations below:

$$Cov\left[X_j, X_k\right] = \frac{1}{n-1}\sum_{i=1}^{n}(x_{ij} - \bar{x}_j)(x_{ik} - \bar{x}_k). \tag{4}$$

$$Matrix(Covariance) = \frac{1}{n-1}\left((\mathrm{X} - \bar{\mathrm{X}})^T(\mathrm{X} - \bar{\mathrm{X}})\right) \tag{5}$$

Where $\bar{\mathrm{X}}$ represents the mean vector, and it can be calculated using the following equation: $\bar{\mathrm{X}} = \frac{1}{n}\sum_{i=1}^{n} x_i.$

The mean vector (d-dimensional vector) stores the mean of each feature column in the dataset.

(3) Eigenvalues and Eigenvectors Calculation

Next, we need to calculate the eigenvalues and eigenvectors for the covariance matrix. Covariance matrix is a square matrix, so the eigenvalue can be calculated using the characteristic equation below:

$$det(\mathrm{D}I - A) = 0 \tag{6}$$

Where, D represents the eigenvalue for matrix A, I is an identity matrix which has the same dimension as A to satisfy the requirement of matrix subtraction. $'det'$ is the determinant of the matrix. We can find a corresponding vector V for each eigenvalue D by solving the following equation:

$$(\mathrm{D}I - A)V = 0 \tag{7}$$

(4) Principal Components Selection

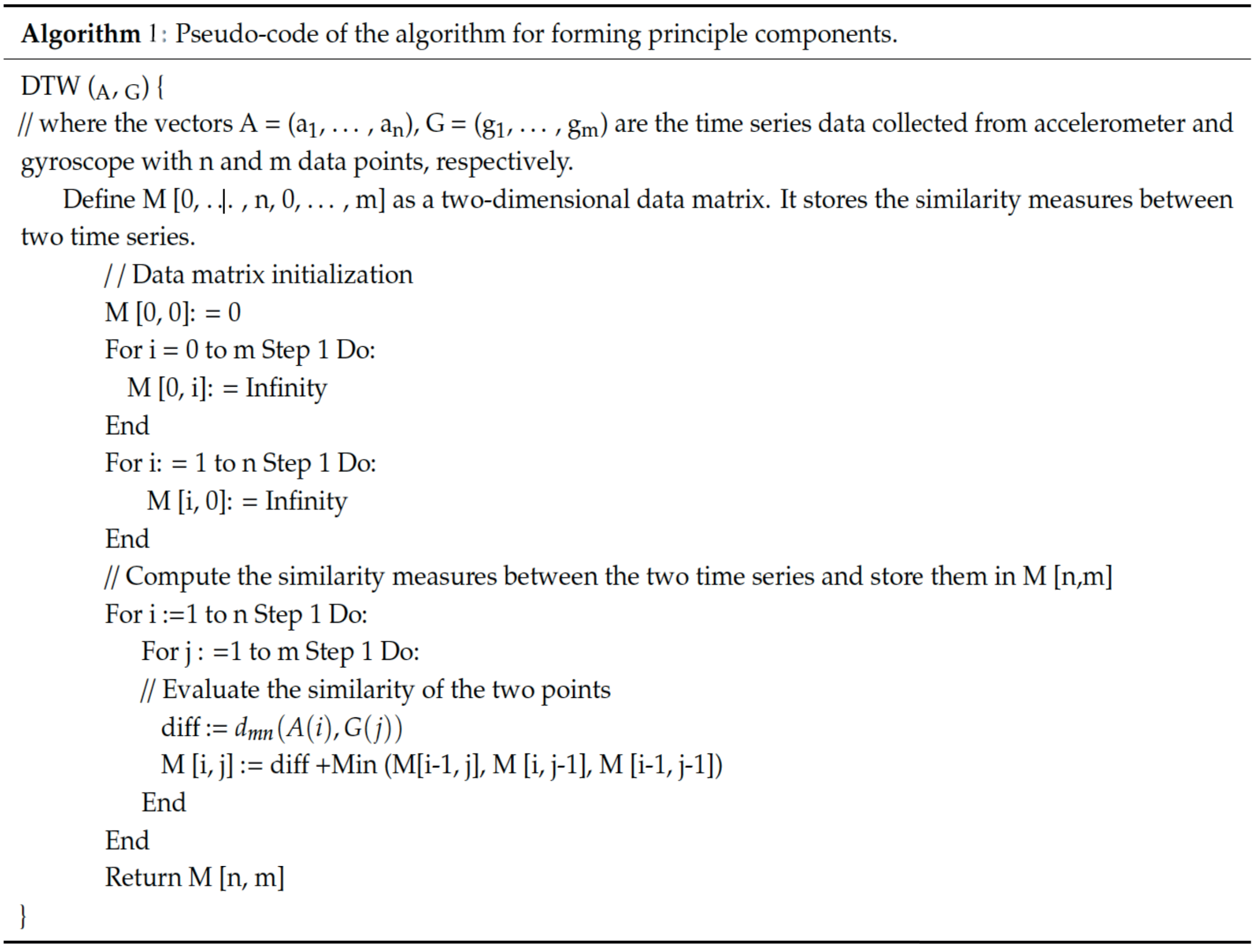
Eigenvalues are sorted in descending order such that it reflects the significance of the components. The eigenvector that has the highest eigenvalue is the principal component of the dataset. Given that our dataset contains two variables (accelerometer and gyroscope), we should have 2 eigenvalues and 2 eigenvectors. We use a feature vector (V_1, V_2) to store the 2 eigenvectors.

(5) Principle Components Formation

The eigenvectors represent the direction of the principal components. The original data needs to be re-oriented to the new coordinate system using the eigenvectors. To re-orient the data, the original data was multiplied by the feature vector. And the re-oriented dataset is called a score (as shown in the following equation).

$$Sc = [Orig.data] \times [V] \tag{8}$$

As discussed in previous sections, we build an event library using motion data collected by different robots. The motion data contains all the typical events. When a new event is detected, the corresponding motion data will be processed using the PCA algorithm and compared with all the pre-defined events in the event library. We use the DTW algorithm to evaluate the distance between the new event temporal sequence and all temporal sequences in the library. Based on the derived distance, a k-Nearest Neighbor algorithm is used to predict a label for the new event.



Algorithm 1: Pseudo-code of the algorithm for forming principle components.

```
DTW (A, G) {
// where the vectors A = (a_1, . . . , a_n), G = (g_1, . . . , g_m) are the time series data collected from accelerometer and gyroscope with n and m data points, respectively.
    Define M [0, . .|. , n, 0, . . . , m] as a two-dimensional data matrix. It stores the similarity measures between two time series.
        / / Data matrix initialization
        M [0, 0]: = 0
        For i = 0 to m Step 1 Do:
          M [0, i]: = Infinity
        End
        For i: = 1 to n Step 1 Do:
            M [i, 0]: = Infinity
        End
        // Compute the similarity measures between the two time series and store them in M [n,m]
        For i :=1 to n Step 1 Do:
            For j : =1 to m Step 1 Do:
            // Evaluate the similarity of the two points
              diff := $d_{mn}(A(i), G(j))$
              M [i, j] := diff +Min (M[i-1, j], M [i, j-1], M [i-1, j-1])
            End
        End
        Return M [n, m]
}
```

To evaluate the distance between two vectors, we have used the following distance metrics.

Euclidean Distance – the root sum of squared differences:

$$d_{mn}(\mathrm{X}, \mathrm{Y}) = \sqrt{\sum_{k=1}^{K} (x_{k,m} - y_{k,n}) * (x_{k,m} - y_{k,n})} \tag{9}$$

Manhattan Distance – the sum of absolute differences, also known as the Manhattan, city block, taxicab, or 1_1 metric:

$$d_{mn}(\mathrm{X},\mathrm{Y}) = \sum_{k=1}^{K} \left| x_{k,m} - y_{k,n} \right| = \sum_{k=1}^{K} \sqrt{(x_{k,m} - y_{k,n}) * (x_{k,m} - y_{k,n})} \tag{10}$$

Squared Distance – the square of the Euclidean metric:

$$d_{mn}(\mathrm{X},\mathrm{Y}) = \sum_{k=1}^{K} (x_{k,m} - y_{k,n}) * (x_{k,m} - y_{k,n}) \tag{11}$$

Symmetric Kullback – Leibler Metric Distance – Only valid when X and Y are real and positive numbers:

$$d_{mn}(\mathrm{X},\mathrm{Y}) = \sum_{k=1}^{K} (x_{k,m} - y_{k,n})(\log x_{k,m} - \log y_{k,n}) \tag{12}$$

3.2.3 Sensing Algorithm Design and Implementation

Using inertial sensors to detect various types of events is accurately and energy efficient. Given the fact that a robot machine should have minimal operations if the machine is expected to do long term work because of the battery limitation. Hence, it is better not to use the camera monitoring the machine during the whole trip, we only need to focus on some unusual objects when an unusual environment is recognized. Based on this fact, our algorithm leverages inertial sensors to detect robot movements and save corresponding video clips only when an unusual motion pattern is detected. Doing this can reduce the computing overhead of the whole algorithm and decrease the energy consumption for the whole system.

The system is always activate to detect “evens” in near real-time, however, there is a small latency since the time series sensor data is analyzed per segment. This segment size is determind when we train the pattern recgnization model. In another word, the segment size varys depending on different robot vehile sizes and speeds. For example,

3.3 Experiment Design and Data Association

3.3.1 Experiment Scenario

In this study, we want to test our proposed algorithms and have precise control on knowing the terrain. So a simulation was built using a sand table that could easily create different terrain conditions using a 1:18 scale. The sand table is a narrow rectangle which is like a downscaled racing track. During the data collection, we counted each run from one end of the sand table to the other end as one instance. To augment the datasets, we collected data with both ends as starting points and the other ends correspondingly as end point. The different setups for each kind of terrain conditions as shown in Table 5 were chosen for the experimental tests. These simulated field conditions were mainly including dry flat path, slope, continuous hills, depression and muddy path. Each scenario was set up on the sand table. The distance of these five experiment scenarios were approximately 2-3 m. Table 5 shows detailed information about each scenario.

Table 3 Indoor experiment scenarios

Category	Distance (m)	Description
Flat	2	Flat board with a layer of soil
Single slope	2	Wood chip covered; Angle 30 degree
Continuous hills	3	Wood chip covered; Angles range from 15-45
Depression	2.5	5-8cm irregular shapes
Muddy	2.5	With small water pit and mud

3.3.2 Hardware Configuration

System implementation: Processing motion sensor data in real-time is very important for our method. Additionally, capture the associated environment images in time also offers accurate reference data for future analysis. Last, we also plan to add a real-time image processing function in future research. To achieve the best performance and extensibility, we choose an embedded computing platform and optimize the inference engine as the hardware container. Our inference engine is optimized to run on this specific embedded computing platform.

Data processing platform: In this work, we choose the Nvidia Jetson TX2 embedded computer as the embedded computing platform. Because we leverage the system with a scale-down prototype environment, we use an external wire as power supply. The Jetson TX2 has a hexcore ARMv8 64-bit CPU complex and a 256-core Pascal GPU. Our system is built upon the multi-thread framework, written in C and C++. We use independent pipelines to manage different tasks, e.g., we create a pipeline to collect data from motion sensors and monitor various events and implement another pipeline to collect environment images. Each pipeline consists of a series of elements. The element is where a data stream is processed. The hardware platform for small agricultural machine development, size and price are the two main parameters that must be considered in the design idea. In this study, we use a small-sized and low-cost IMU sensor to capture acceleration and angular speed change.

IMU sensor selection: We applied a 6-axis IMU and environmental sensor during our data collection. This sensor named MetaTracker (MTR) is a precise device that offers real-time and continuous monitoring of motion and environmental sensor data. A waterproof and shock absorbent case can be mounted to walls, ceilings or dropped in a package. The sensors are ideal for vibration detection, object movement, and robotics. The sample rates are 0.001Hz – 100Hz stream – 800Hz log for both gyroscope and accelerometer.

Camera selection: Specifically, we choose the Logitech-C922x-Pro USB camera in this work with frame rate of 15fps. Logitech C-series USB cameras are widely chosen as monocular cameras when building prototype systems.

3.3.3 Data Collection

Data collection system design: A robot machine usually has limited battery capacities, so it is important to reduce unnecessary operations as much as possible if the machine is expected to do long-term work. Hence, it is better not to use the camera monitoring the machine during the whole trip, we only need to focus on some specific environmental images when an unusual topography is detected. Keeping this in mind, our algorithm leverages inertial sensors to continuously detect robot vehicle movement and only collecting corresponding camera data once an unusual motion pattern is detected. Given the facts that IMU sensors consume much less power than of cameras, this in return can dramatically decrease the energy requirement and the computing overhead of the whole algorithm.

Historical data collection: Because we need enough historical data which includes all kinds of scenarios with different terrain changes, most data collection effort was spent in this part of data collection. Also, the whole datasets were divided into two parts, one for training the pattern detection model and the second part for use with the trained model to test its accuracy. The ratio between training datasets and testing datasets is flexible and depends on the amount of data collection. In our work, we used a ratio around 2:8 for training datasets and testing datasets.

Realtime data collection: After we build a confident algorithm with specific parameters, we conducted additionally real-time testing. In this case, we also collected some additional data for leveraging. A real-time data streaming was generated on a local machine and tested by the algorithm.

3.4 Result and Discussion

We demonstrate and discuss the performance of our approach in recognizing dry flat surface field, slope, continuous hills, depression and muddy field separately. In each scenario, a motion pattern and corresponding topological feature are plotted and discussed together in the following subsections. For different scenarios, we applied different speed setups for better quality data, and we also made some data processing, like normalization, denoising and etc.

(1) Single Slope

As is shown in Figure 23, the field slopes were estimated using a gyroscope via coordinates alignment. The downslope, with a sharply increasing, and upslope, with a deep decrease, are also marked. The derivative could easily address the gradient of the slope. If the gyroscope value climbs up fast then drops down quickly, the slope is steep. While the slope has a minor gradient if the gyroscope value waves not obvious. The results in Figure 23 give us an accordant pattern of slope detection in practice to the theoretical pattern. If, in some cases, the collected data couldn't perfectly match the pattern. In those cases, the estimated linear acceleration was calculated using the accelerometer to reduce noise in the sensor data as shown in Figure 24. However, in most cases, this single slope can be recognized by only using the gyroscope data.

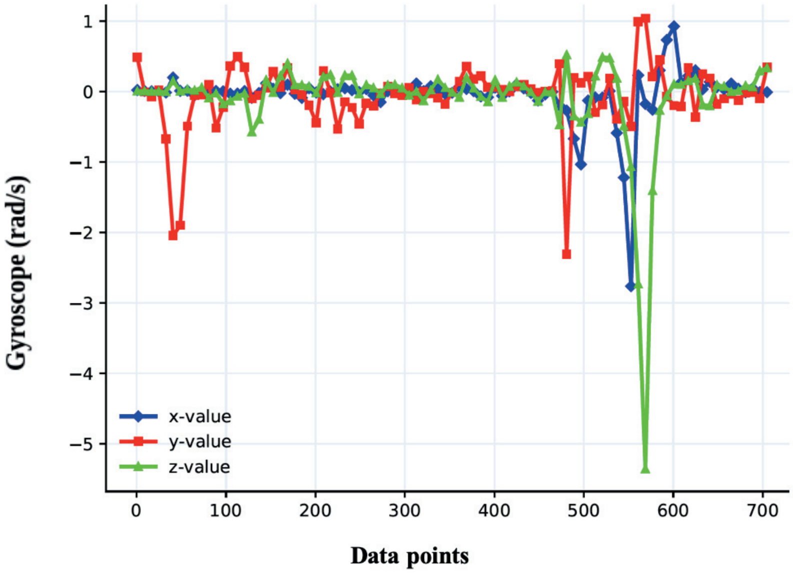


Figure 23 Estimate path slopes with Gyroscope

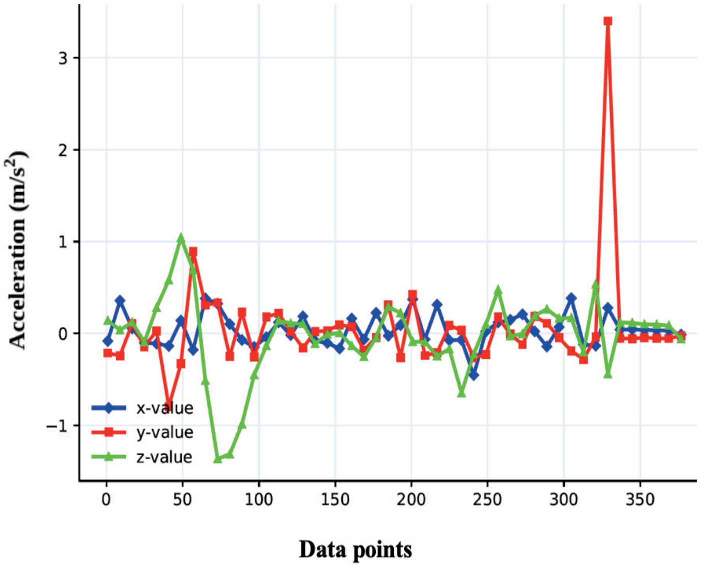


Figure 24 Estimated linear acceleration

(2) Continuous Hills (lunar landscape)

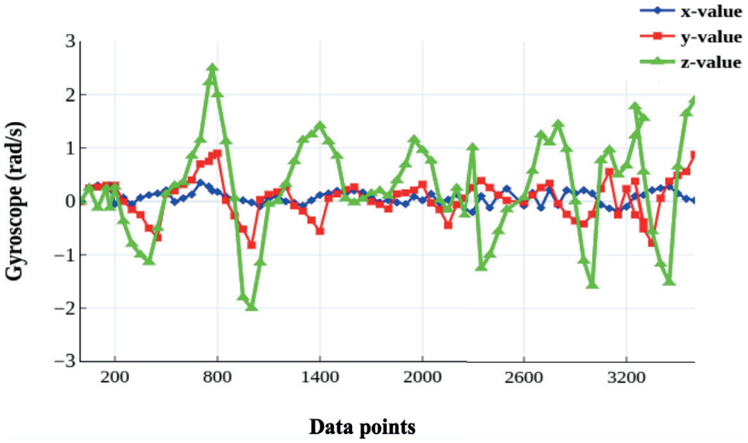


Figure 25 The segmental Gyro data of continuously hill up/down

A continuous hills landscape was indicated by a continuous occurrence of small hills or lunar landscape, which is commonly seen in the farmland. As is shown in Fig. 25, the readings of z-axis have a sine or cosine wave pattern, and each upper peak wave is following a bottom peak wave, which indicated the robot going up and down hills. According to this pattern in this Fig. 26, there are four complete uphill and downhills traversed.

(3) Depression or Soil Erosion

A depression can be identified from changes of values along z-axis as shown in Fig. 27. If the z-axis readings approximately stable, then it should be a smooth path without bumps or depressions. If the readings start fluctuating, that indicates there might exist a bump or depression. If the magnitude of readings along z-axis changes from positive to negative, and readings of x and y axis also changes, then it is very likely there is a depression.

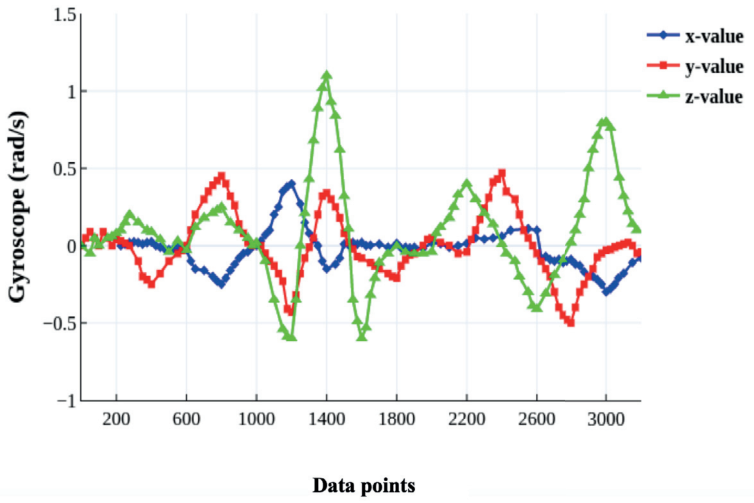


Figure 26 The segmental Gyro data with meeting a depression

(4) Rough Field

The rough field is indicated by a continuous occurrence of bumps or depressions as shown in Fig. 28. If continuous peaks and valleys can be observed from z-axis values, then it is highly likely that there is a rough patch of the path. It can be divided into several depressions, which means the pattern of the rough path is the combination of depression patterns.

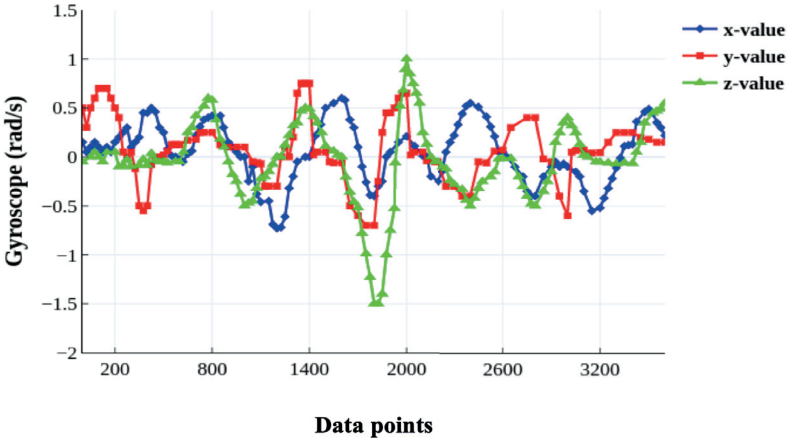


Figure 27 The segmental Gyro data of the rough field

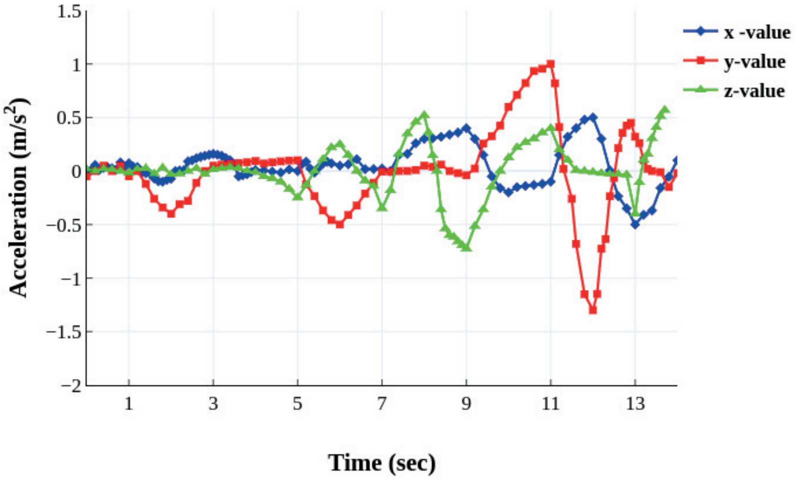


Figure 28 Deceleration happens with path surface change

(5) Flat Path vs Muddy Path

A muddy path could cause sudden deceleration as is shown in Fig. 28, where the muddy path starts from 9ms. The deceleration can be inferred from changes in readings along the y-axis. As we can see from this example, there is a deep valley at around 12 seconds, and the readings of both x and z axis are also fluctuating.

3.5 Conclusions

Although multiple sensors are used to augment the insight of the 3D map, while the algorithm still uses a single camera, which is not sufficient for farm landscape measurement. One possible improvement to our approach is to leverage distance sensors for gathering landscape information for 3D map construction. This chapter presents the sensing algorithm, a low-cost, robot-mounted, multidimensional map construction method that can track robot movements, monitor the surrounding environment, and link all the factors to the 3D map, thereby providing useful analytics to task planning, route planning and robot operators. The method leverages IMU sensors to gather mobility data for every individual robot. In this work, we attempted to provide analytics and data fusion from several specific aspects of the robot working environment. We believe our farmland sensing approach has many more interesting and useful applications in similar agriculture environments.

Chapter 4. Perceiving: Agriculture Object Detection based on Computer Vision

This chapter has been submitted to the journal Remote Sensing entitled "Augmenting Crop Detection for Precision Agriculture with Deep Visual Transfer Learning – A Case Study of Bale Detection."

4.1 Introduction

According to the United Nations population estimates and projections, the growing world population will be over 10 billion in 2050 [2]. Correspondingly by 2050, there will be an increase in food demand by around 59% [154] since the significantly increased population. To increase crop production while minimizing inputs, the adoption of advanced computing technologies, including computer vision, machine learning, and big data analytics have recently gained interests among researchers in the fields of agriculture. Smart agriculture takes advantage of advanced computing technologies to minimize the inputs required, to improve the crop quality and to increase yields.

With the reduction of equipment costs, increase of computing power, and availability of non-destructive food assessment methods, the efforts of many researchers and practitioners to improve the crop quality and yields have focused on computer vision and machining learning [14]. Computer vision helps with object detection and machine learning allows useful information that can be extracted from the collected data to be available, showing tremendous advantages over the traditional methods applied in agriculture [15].

Other research efforts have shown that the combination of computer vision and machine learning techniques on the multiple periods of crop production and harvesting are promising [16]. Computer vision in agriculture can be applied easily to analyze digital images collected from the fields and to provide high-level understandable information to the users [17]. For example, computer vision not only detects the weeds fast and effortlessly, but also accurately applies treatment with the help of a ground robot [18]. In addition, computer vision can detect the diseases on the crops and inform users for them to then take action [19, 20].

During the harvest process, the logistics of biomass aggregation and transportation is essential. For example, the United States has significant lignocellulosic biomass [21] resources that could be used for emerging industries like biofuels. However, converting the biomass to renewable energy is not currently economically feasible with more efficient collection methods needed [22]. With the application of computer vision and machine learning, baled biomass can be detected accurately and fast, benefiting the harvest process by allowing for improved collection routes and yield determination. Although this is just one example it demonstrates how this technology can play a significant role in the crop harvest process.

Image Acquisitions. To collect the images as inputs for the computer vision, using Unmanned Aerial Vehicle (UAV) is an efficient approach, which has been widely used in precision agriculture as well as many other fields, such as path planning, design, and livestock detection [23,24]. UAV combined with computer vision can also contribute to remote sensing to help inform farmers about the geo-specific crop yield and identify crop diseases [25, 26]. Sometimes, decisions are required to be made off-board once the data have been collected and processed by the UAV, based on the information provided by the images processed from the computer vision technique [27] For example, UAVs can be used to detect a potential issue, and then obtain high-resolution images or inspect and apply treatments correspondingly.

Bale detection challenges. When it comes to object detection, associated methods are commonly sensitive to the illumination, and object and background domain change. A non-robust model can easily fail if it does not take into account the variation of the light condition [155, 156]. Because of the diversity of illumination situations, seasons, and weather conditions, object detection in the outdoor environment is more complicated than that in the indoor environment since humans can manipulate a consistent environment as is shown in Table 6.

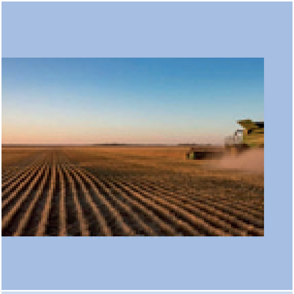
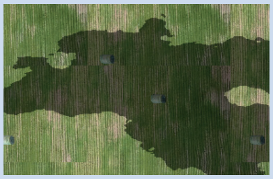
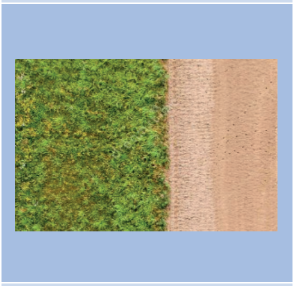
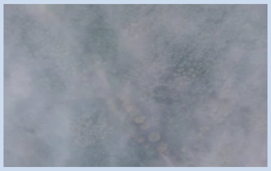
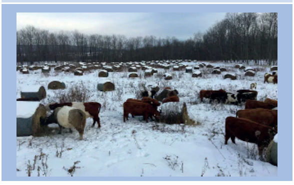
To emphasize, the illumination and hue change are the most significant reasons that impact the bale detection model performance. Illumination variation, including the change of light conditions, and with/without shadow covering, plays a significant role in object detection in the context of outdoor practices. Patrício and Rieder [17] suggested that consistent light conditions between the source domain and target domain will decrease the difficulties of shaping accurate classification models built on the deep learning architecture. A similar conclusion has been drawn by Hornberg [157] that adequate lighting in the environment can increase the reliability of the performance of the models based on the collected images.

Hue change due to the season transition and variation of light conditions is another key factor to be considered in precision agriculture. During growing season of vineyard when the lights are not strong enough, Baweja et al. [158] added extra light when collecting images by using strobe lighting mounted on a ground robot image capturing machine to compensate the hue variation to build a reliable deep learning model.

Since the deep learning-based object detection model always needs a large number of images labeled as the ground truths before training a supervised object detection model, the accuracy of detecting performance is impacted by labeled data quality. One approach to improve the quality of labeled data is to include balanced data to include various images from the target domains, listed in Table 6. However, if we want to guarantee the quality, that means for each condition, the total number of images required to be manually labeled could be large and take significant resources to complete.

To reduce the task of labeling the objects manually, style transferring methods have been developed. To minimize the discrepancy between the source domain and target domain regarding the domain distribution, we propose a model by combining the Convolutional Neural Network (CNN) based YOLOv3 model and domain adaption (DA), a representative method in transfer learning. Domain adaption works very well where the tasks are similar except for the domain distribution between the source domain and target domain [159]. Fig. 29 illustrates the proposed biomass detection model on the basis of CNN and DA. Since it has strengths of accuracy and speed for object detection, the YOLOv3 was selected to build the CNN model [160]. To realize the DA, the unpaired translation method, Cycle Generative Adversarial Networks (CycleGAN), was used to tackle the image difference due to the illumination, hue and Clarity discrepancy.

Table 4 Object detection challenges in complex unconstrained outdoor environments

Environment diversity	Conditions	Example	Agriculture Information	Technical challenges for bale detection
Illumination diversity (Target domains-1)	Lighting condition		To gain the efficiency of agriculture, different process routines are conducted to crops in morning, afternoon and night.	Decreasing the difficulties of shaping an accurate classification models built on the deep learning architecture.
	Shadow		Shadow is commonly seen during daytime. This always happens in rain season. The images taken by UAV includes shadows in certain months.	Shadows cross the objects decrease the accuracy of the classification on these kinds of objects. Also, the scale of background and bale size make it worse.
Seasons change (Target domains-2)	Hue change		Farms with different plants have various harvest seasons. As a result, the bales and backgrounds vary in different season.	The inconsistency changes of background and bale with season trigger the decrease of bale detection performance.
Adverse Weather Condition (Target domains-3)	Haze		Haze weather sometimes happen with temperature drop or precipitation change. This may cause the grain lifecycle adjustment, that need to be monitored.	The high performance of supervised learning and semi-supervised learning – (object detection) in haze weather is always a challenge.
	Snow covered		Tracking bales in winter and snow environment is also important for continuously feeding livestock.	Restoration-based algorithms may mislead or overfit the object comparing to the original one. The snow weather reduces the features of objects in the images.

4.2 Methodology

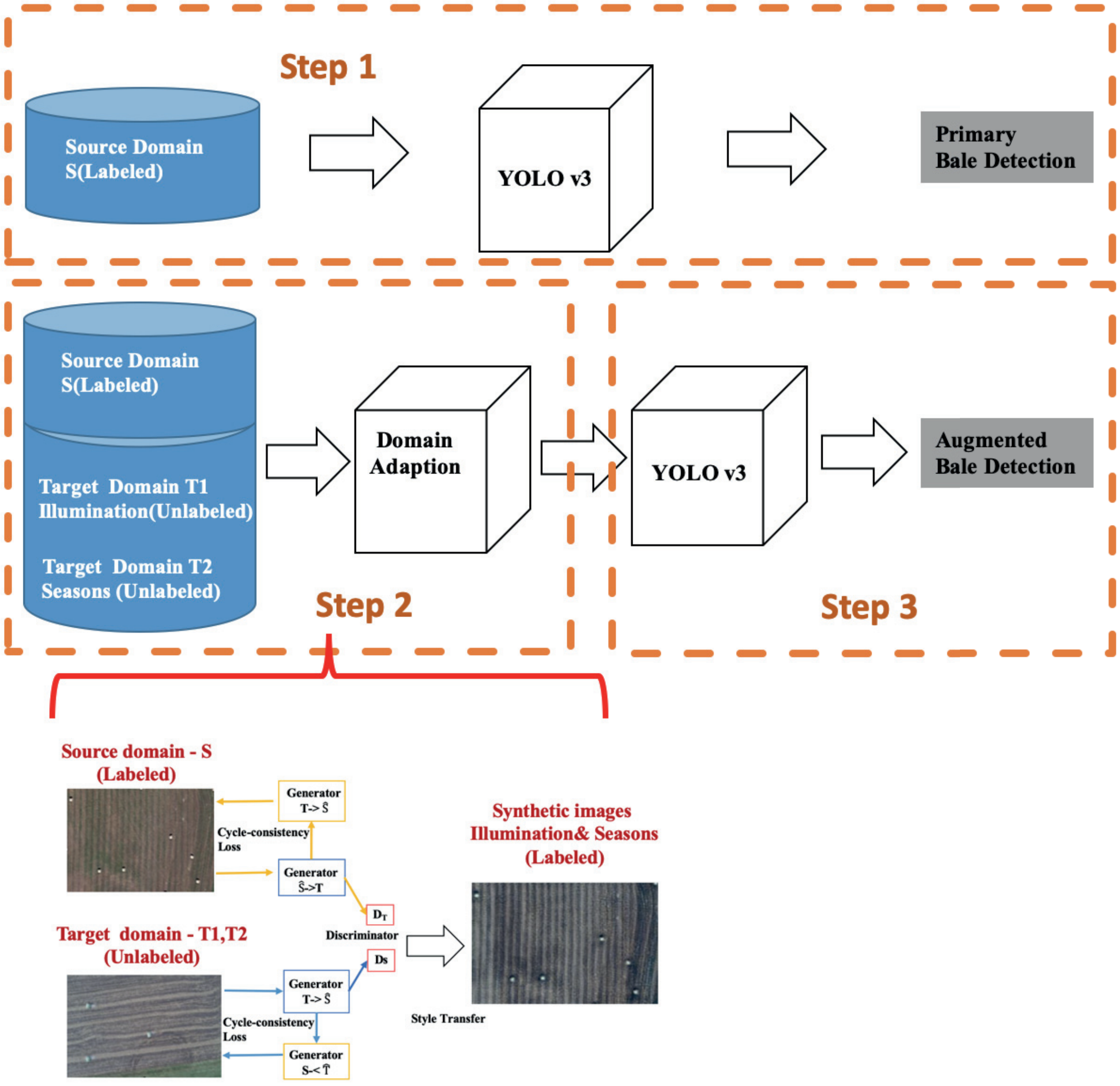


Figure 29 Framework summary of proposed bale detection method pipeline

Bale detection method pipeline summary

Fig. 29. shows the completed structure of the bale detection method, from image acquisition to creating the model, then to augment the model, proposed in this work. We divide this pipeline into 3 steps as follows. Step 1 trains a primary object detection model with the YOLOv3 only based on the manually labelled initial condition images. Step 2 demonstrates the method how we use manually labelled ground truth images to generate more ground truth images with automatic labels. Then in step 3, we augment the object detection model in step 1 with mixed labelled ground truth images as training data.

Step 1: Primary Object detection

A YOLOv3 model is trained for primary bale detection using 243 images captured with good illumination conditions in fall. We define these labelled images as source domain. CNN based object detection methods, such as Faster R-CNN, YOLO, and Mask R-CNN, gained the popularity among researchers and have been proved to be efficient [160, 161, 143]. YOLOv3 was released by Redmon and Farhadi in 2018, extended from the previous YOLO versions [162]. In this paper, YOLOv3 is implemented in the baled detection process, taking advantages of its accuracy and fast speed on object detection. Instead of using multiple networks for analysis, YOLOv3, indicated by its name You Only Look Once, passes the input image once to a convolutional neural network, lowering the costs and improving the performance significantly. In addition, the network splits the input into multiple regions and works on each one with the bounding boxes and their classification probabilities. By focusing on the global context of the image, YOLOv3 decreases the possibility of making location classification error.

To implement YOLOv3, we used PyTorch to train the model and to make inferences, based on the Darknet-53 (an architecture consists of 53 convolutional neural networks). The initial weights between layers were provided by the Darknet-53 backbone [162]. YOLOv3 relies on a deeper architecture to extract features, the backbone here is “Darknet-53” with 53 convolutional layers. Leaky ReLU activation as well as normalization are added to every layer. Instead of using any form of pooling often contributing to loss of low-level features, we applied a stride of 2 in convolutional layers to reduce the size of samples of the feature maps. Stride refers to the factor between the applications of the filter to the input image. An image of size 416x416, for instance, can be down-sampled to 13x13 by a stride of 32. The shape of input images is (m, 416, 416, 3). The output consists of bounding boxes, representing the recognized classes. Each bounding box is defined by 6 numbers ($p_c, b_x, b_y, b_h, b_w, c$). With augmenting cc (class) to an 80 dimensions vector, 85 numbers are used to describe every single bounding-box as shown in Fig. 30.

Similar to object detectors, features learned by the convolutional layers are filtered to predict detection, such as the coordinates of the bounding boxes and the class label. YOLO v3 is based on 1 to 1 convolution to predict so the prediction map has the same size as the input. Each cell in the prediction map represents a fixed number of bounding boxes as shown in Fig. 31.

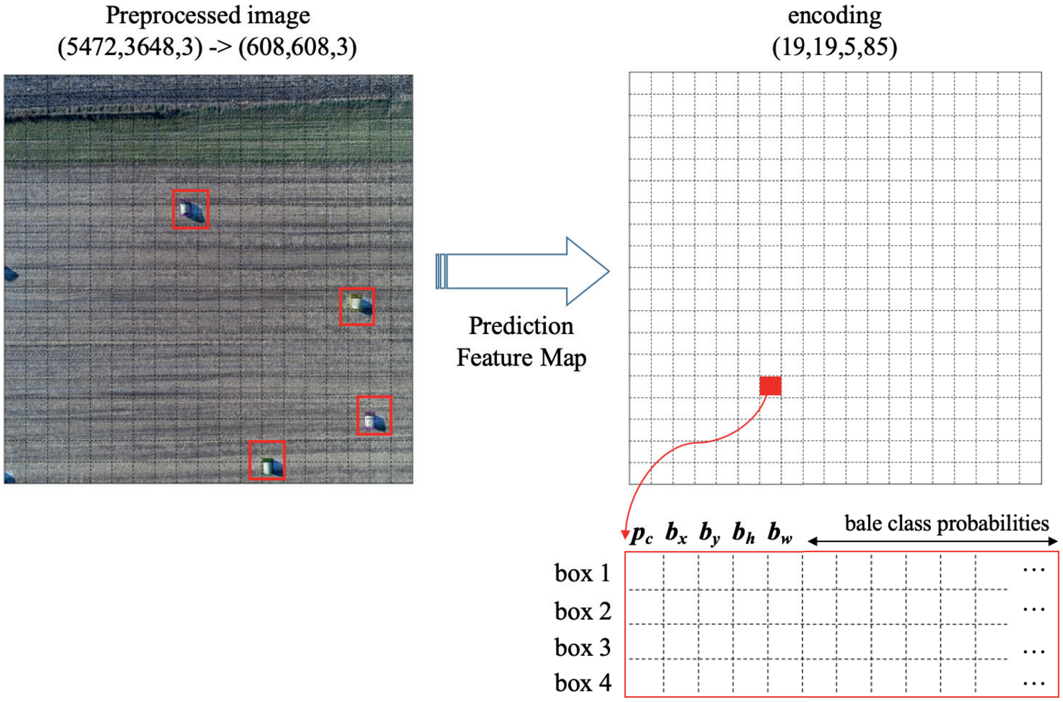


Figure 30 Explanation of encoding in YOLOv3 architecture

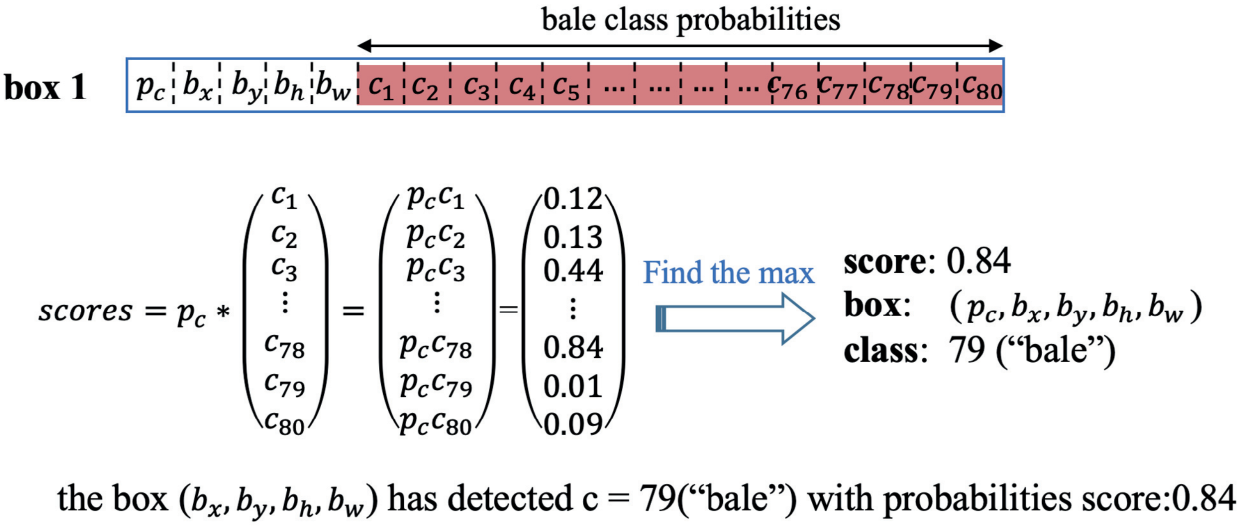

bale class probabilities
box 1
p_c b_x b_y b_h b_w c_1 c_2 c_3 c_4 c_5 … … … … c_{76} c_{77} c_{78} c_{79} c_{80}
$scores = p_c * \begin{pmatrix} c_1 \\ c_2 \\ c_3 \\ \vdots \\ c_{78} \\ c_{79} \\ c_{80} \end{pmatrix} = \begin{pmatrix} p_c c_1 \\ p_c c_2 \\ p_c c_3 \\ \vdots \\ p_c c_{78} \\ p_c c_{79} \\ p_c c_{80} \end{pmatrix} = \begin{pmatrix} 0.12 \\ 0.13 \\ 0.44 \\ \vdots \\ 0.84 \\ 0.01 \\ 0.09 \end{pmatrix}$
Find the max
score: 0.84
box: $(p_c, b_x, b_y, b_h, b_w)$
class: 79 ("bale")
the box (b_x, b_y, b_h, b_w) has detected c = 79("bale") with probabilities score:0.84

Figure 31 Example of generating probability of certain class in each bounding box

Step 2: Augmenting training data with Domain Adaptation

Domain Adaptation as one kind of Transfer Learning is designed for augmenting the training data scenarios with automatic labels. As shown in lower left in Fig. 31, more than two more conditional images are list as Target domain 1, 2, etc. Traditionally, all the targeting objects in the images need to be manually labelled. However, our proposed method, combining the YOLOv3 with DA, decreases the laborious manual identification work but also ensure the performance of the model by applying style transferring.

We only labelled the inputs from the images with one condition and then we collected limited more image with diverse illuminations, hues, and styles under different environments. Then we built a domain transferring model to convert the images of initial condition to new images of the other conditions. Instead of manually labelling all the inputs required by the model, only part of the images was manually processed and the rest of the inputs shared the same label automatically because of the style transfer. In this way, a more robust YOLOv3 model which performs accurately on augmented styles of images could be achieved.

The DA technique is applied to shape the translation mapping from the source-domain (S) in the initial environment to the target domain T in the other environments, and vice versa. Images from two different domains are not related in any way. CycleGAN [163] is implemented to transfer the styles between two domains to synthesize target domain images from the source-domain (S).

Two GANs are used for applying the CycleGAN in style transfer. Each one includes one generator and one adversarial discriminator. The generator, $Gen_{(S,T)}$, in the first GAN translates images from the source-domain (S) to the target-domain (T), while the adversarial discriminator D_T outputs the likelihood that the images taken from the target-domain (T) are real images. Similarly, the generator $Gen_{(S,T)}$, in the other GAN translates images from the target-domain (T) to the source-domain (S), and its adversarial discriminator D_S outputs the likelihood that the images taken from the source-domain (S) is real images from the source-domain (S). I_S and I_t represents images from domain (S) and (T), respectively. Given $i_S \in I_S$ and $i_t \in I_t$, is and it represents images in domain (S) and (T), respectively.

$\hat{T}$ represents the domains of images synthesized in Fig. 31. It represents the domain for the diverse season and illuminations synthetic images generated from the real initial environment images, while $\hat{S}$ denotes the initial synthetic images generated from the real other environment images. By applying $Gen_{(S,T)}$, images $i_S \in I_S$ is transferred to synthetic-images in $\hat{T}$, while the corresponding adversarial discriminator improves model by encouraging the translated image hardly distinguishable from the domain (T). Ideally, when the translated image from the source-domain (S) to the target-domain (T) is translated back from the target-domain (T) to the source-domain (S), we should get identical images. However, learning models are not perfect and two different images will be obtained. The difference between the two images is measured by cycle consistency loss, which is defined as below by following (2):

$$\begin{aligned} Ls_{Cycle}\left(Gen_{(S,T)},Gen_{(T,S)},D_S,D_T,S,T\right) = \lambda Ls_{Cycle}&\left(Gen_{(S,T)},Gen_{(T,S)},S,T\right)+ \\ Ls_{GAN}&\left(Gen_{(T,S)},D_S,T,S\right)+ \\ Ls_{GAN}&\left(Gen_{(S,T)},D_T,S,T\right) \end{aligned} \tag{13}$$

In Equation (13), λ is the balance weight, Ls_{Cycle} measures the cycle consistency loss, Ls_{GAN} represents the loss function of the adversarial training. The cycle consistency loss used in GAN training penalize Ls_1 in the cycle architecture, defined as:

$$\begin{aligned} Ls_{Cycle}\left(Gen_{(S,T)},Gen_{(T,S)},S,T\right) = \mathrm{E}_{i_T\sim I_T}&\left[\left\|Gen_{(T,S)}\left(Gen_{(S,T)}\left(i_S\right)\right)-i_S\right\|_1\right]+ \\ \mathrm{E}_{i_T\sim I_T}&\left[\left\|Gen_{(S,T)}\left(Gen_{(T,S)}\left(i_T\right)\right)-i_T\right\|_1\right] \end{aligned} \tag{14}$$

Equation (15) defines the loss in adversarial training:

$$Ls_{Cycle}\left(Gen_{(S,T)},Gen_{(T,S)},S,T\right) = \mathrm{E}_{i_T\sim I_T}\left[\log\left(D_T\left(i_T\right)\right)\right]+\mathrm{E}_{i_T\sim I_T}\left[\log\left(1-Gen_{(S,T)}\left(i_S\right)\right)\right] \tag{15}$$

To train these generators and discriminators, we need to solve:

$$\begin{matrix} Gen_{(S,T)}{}^{*} \\ Gen_{(T,S)}{}^{*} \end{matrix} = \arg\min_{Gen_{(S,T)},Gen_{(T,S)}} \max{}_{D_S,D_T} Ls_{CycleGAN}\left(Gen_{(S,T)},Gen_{(T,S)},D_S,D_T,S,T\right) \tag{16}$$

Gradient descent is first applied to Equation 16, followed by backpropagation to allow the generator $Gen_{(S,T)}$ complete style transfer between real initial-style images and synthetic other-style images without changing the spatial relationship between the biomass in the images.

Step 3: Optimize YOLOv3 model with extended datasets from Step 2.

There are two optional methods we can apply to optimize the performance of the model. One is retraining the model, and other one is fine-tuning. Retraining a model using extended data with a proper pre-processing is a straightforward and robust way, however it takes longer time than fine-tuning.

A commonly used manner to transfer trained model to the new dataset is fine-tuning, which is more efficient when the size of the new dataset is small. Fine-tuning trained models can not only reduce the probability of overfitting, but also provides better generalization if the original dataset and new dataset share similar domains. In this research, we applied both methods keeping the better results of the two.

4.3 Experiment Design and Data Association

4.3.1 Experiment Equipment

The input data with baled housing biomass were collected from the fields by a drone from the Arlington Research Station (Arlington, WI, USA) The drone, equipped with a 1-inch Exmor R CMOS sensor and a gimbal stabilizer which handles the lateral and vertical vibration, allows us to collect images from different heights as shown in Fig. 32. Through each campaign, the locations of baled biomass are identified by a Global Navigation Satellite System (GNSS) and their corresponding centers are surveyed by a Carlson Surveyor 2. These two additional systems are for validation of the location accuracy and as a contribution to public database for future research.

Figure 32 Zenmuse-X4S camera equipped DJI-Inspire-2 UAV.

4.3.1 Bales Data Collection and Description

All the following images collected in biomass bales fields are with this model of drone. Images from two different heights, 200 ft and 400 ft, were captured through seven campaigns to provide different resolutions to test our model performance. The size of the collected images is 5472 x 3648 pixels, corresponding to a 20-megapixel resolution as shown in Fig. 33. In addition, we created a second dataset by rescaling the collected images to 1080 x 720 with 3:2 ratio, simulating a camera with less than 1-megapixel resolution. The image numbers specifications used in the experiments are shown in Table 7 as “Initial condition”. There are totally 300 images used for training, validation, and testing. All these images were collected in the fall under good illumination condition without shadow. We also collected 128 real images under the other conditions as ground truths for both training CycleGAN model and testing performance. More images under the other conditions were generated by CycleGAN model.

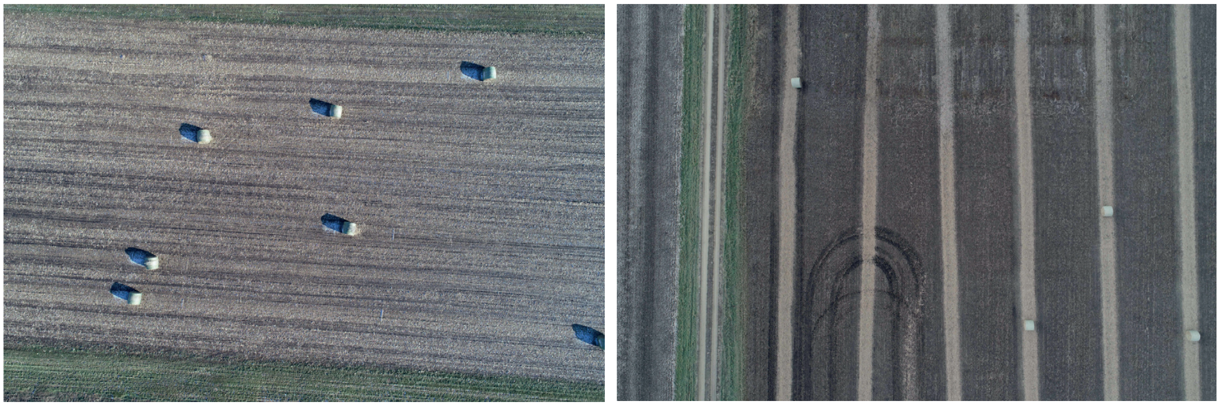


Fig. 33 Example of drone collected images under good lighting condition in fall (Initial condition)

Table 5 Experiment data collection distributed under different environmental conditions (note: there are images cross counted in different conditions)

Environment	**Condition**	**Training**	**Validation**	**Testing**
Initial condition	Good illumination, fall, w/o shadow	243	27	30
Diverse Illumination (Target domains-1)	w/ Lighting condition change	160	20	20
	w/ Shadow	158	20	20
Seasons change (Target domains-2)	Hue change (Summer)	185	19	19
	Hue change (Early winter)	187	12	12
Adverse Weather Condition (Target domains-3)	w/ Haze	159	20	20
	w/ Snow covered	150	19	19

Fig. 33 provides an example of the images used in the model. Baled biomass, and streets in all collected images were annotated in both MS COCO and YOLO data formats by using the Computer Vision Annotation Tools (CVATS) LabelImg and LabelMe [164, 165].

4.4 Result and Discussion

4.4.1 Primary Bale Detection with YOLOv3 Corresponding to Step 1

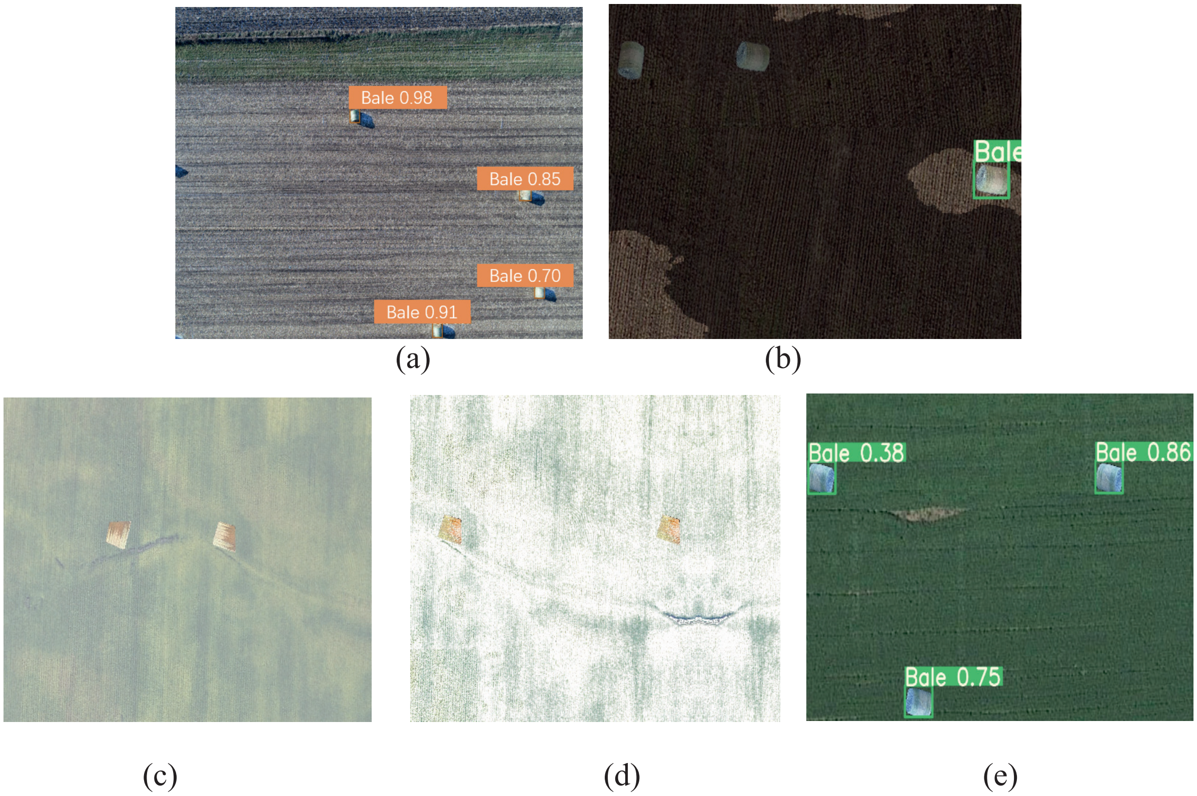
The YOLOv3 detector trained with only initial condition images in Step 1 was applied to detect bales in the real images. Although the training processes don't include images under extended conditions, we still include these images in testing results for comparison with the optimized detection model. The testing results, in terms of precision, recall, mAP, and F1 score for each scenario, are presented in Table 8. The high value of precision indicates a low incidence of false positives, meaning that the algorithm doesn't detect a Bale where doesn't exist. On the other hand, the low recall means the algorithm fails to see some of the bales inside the image.

The prediction performance on initial condition images are excellent, all the precision, recall, mAP, and F1 values are over 0.92. However, these four indices vary in the negative way for the extended conditions. The precision values for all conditions except shadow are over 0.85, which are good. As is shown in Fig. 34(b), bales are commonly failed to be detected inside or across the shadow. The other three indices (recall, mAP, and F1) are all lower than expected for the extended conditions (average values are less than 0.59, 0.7, 0.7 respectively). The F1 score is the harmonic average between precision and recall. Since the last one was low, the F1 score also got low. The mean average precision (mAP) was low and it varies through the different simulated scenarios. However, mAP was high in the haze condition since all the images with haze condition are collected with minor haze or fog, which may cause a significant blur on the background instead of bales. The general results are as expected since there are few image samples in the extended conditions in the training datasets.

Table 6 YOLOv3 model (Trained in Step 1) performance for detecting bales in different conditions without being trained with synthetic images.

Data	Images[1]	Precision	Recall	mAP	F1
All Conditions	148	0.859	0.599	0.780	0.746
Initial condition	30	0.929	0.993	0.987	0.960
Illumination	20	0.881	0.587	0.848	0.735
Shadow	20	0.675	0.456	0.622	0.621
Hue change (Summer)	19	0. 917	0.644	0.853	0.783
Hue change (Early winter)	19	0.929	0.605	0.751	0.852
Haze	20	0.910	0.871	0.975	0.931
Snow	19	0.874	0.456	0.584	0.621

Some examples of tested bale images under multiple environmental conditions using the model trained in Step 1 are shown in Fig. 34. These typical results under different conditions display the same trend with Table 8. There are some undetected bales and low confidence scores listed in Fig. 34.



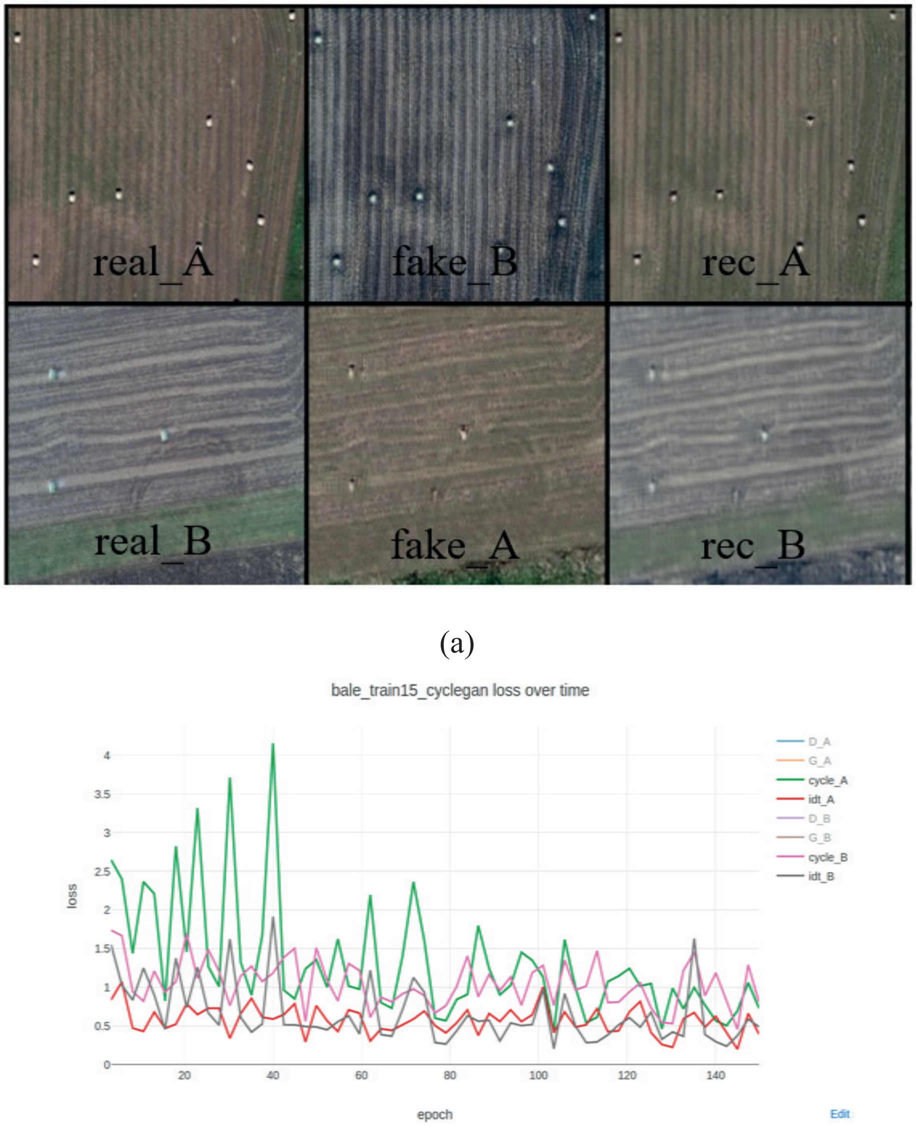
(a) (b)

(c) (d) (e)

Figure 34 Example of tested bale images under multiple environmental conditions using primary bale detection model (trained in Step 1): (a) initial condition with good illumination and no shadow in fall; (b) extended condition – early winter with shadow; (c) summer with haze; (d) winter with snow covered; (e) summer with good illumination condition.

4.4.2 Augmenting Training Data with CycleGAN Corresponding to Step 2

During step 2, we build a CycleGAN model to convert real images to synthetic/fake images as shown in Fig. 35(a). Real_A and real_B are real images, fake_B is the synthetic/fake image from real_A, and rec_A stands for reconstructed image A based on fake_B. The second row has the same idea as the first one. With this CycleGAN model, 1200 synthetic images are generated which will be used as extended training datasets in Step 3.

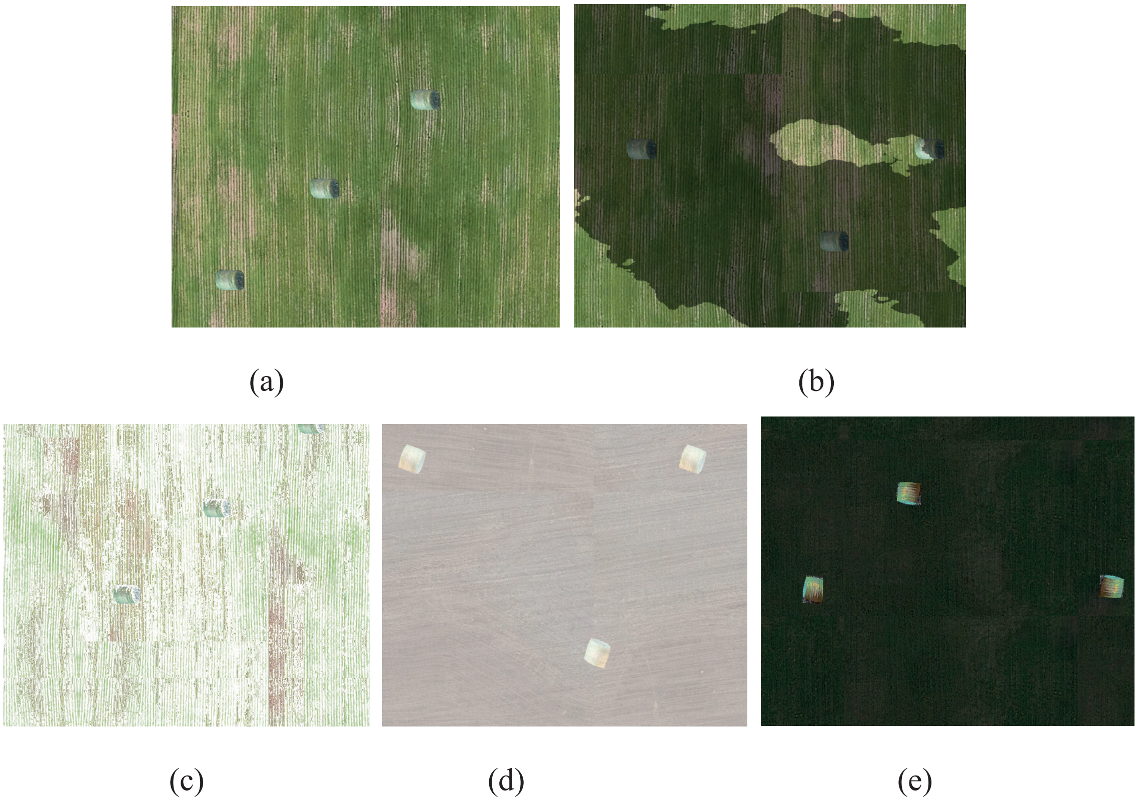


(a)

(b)

Figure 35 Description of CycleGAN model: (a) examples of real images, fake images and reconstructed images; (b) loss tracking during training CycleGAN model.

Identity loss is the index when measuring the discrepancy due to translate one style images to another style images, regulating the generator to generate images with high fidelity translated from the real samples in the target domain. No extra change is needed for the images almost distinguishable from the target domain. Generally, greater identity loss value will be applied for unknown content. Fig. 36(b) shows a slight reduction of the loss, especially the cycle_A in green color, which fits our expectation. More results are displayed in Appendix 1. More information about the model parameters and logic can be found in Zhu et al. [102]. Fig. 36 shows some example of augmented bale images with multiple environmental conditions.



(a) (b)

(c) (d) (e)

Figure 36 Example of augmented bale images with multiple environmental conditions: (a) Summer w/ good illumination; (b) Summer w/ shadow; (c) Winter w/ snow; (d) Early winter w/ haze; (e) Summer w/ dark illumination

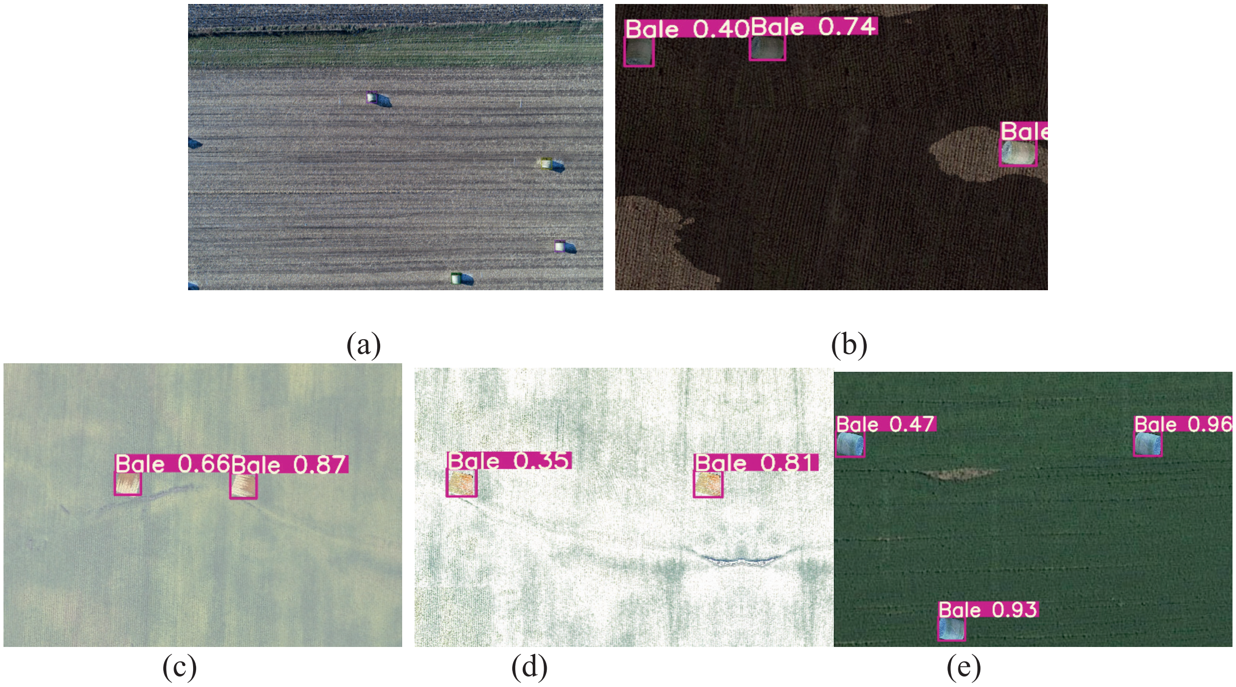
4.4.3 Optimized YOLOv3 Model with Extended Datasets Corresponding to Step 3

Table 7 YOLOv3 model (Trained in Step 3) performance for detecting bales in different conditions after being trained with synthetic images from Step 2.

Data[1]	Images[2]	Precision	Recall	mAP	F1
All Conditions	148	0.869	0.927	0.941	0.892
Initial condition	30	0.913	0.980	0.990	0.945
Illumination	20	0.847	0.926	0.959	0.885
Shadow	20	0.847	0.933	0.854	0.888
Hue change (Summer)	19	0.836	0.933	0.954	0.882
Hue change (Early winter)	19	0.905	0.893	0.969	0.831
Haze	20	0.831	0.867	0.895	0.848
Snow	19	0.926	0.878	0.941	0.901

The optimized YOLOv3 detector trained with both real images and synthetic images in Step 3 is applied to the same testing datasets. Table 9 shows the testing results, which will be compared with the performance of the primary YOLOv3 model in Step 1. YOLOv3 in Step 1 and 3 have a similar performance for bale image detection under the initial condition as shown in the line "Initial condition" in Table 8 and Table 9. The generic testing results, in terms of precision, recall, mAP, and F1 score for each scenario, are presented in Table 9. In most cases, the recall, mAP, and F1 score, are obviously improved from average (0.59, 0.7, 0.7) to average (0.93, 0.94, 0.89) respectively. All the significantly increased values are marked in green. The increment of recall indicates that most of the bales that can't be detected in Step 1 are detected in Step 3. Meanwhile, the precision value keeps a similar level with occasional reduction because of the occasional increased false positives and true positives. This result is a strong evidence that using synthetic images from transfer learning is a reasonable approach to enhance the detection capability with images under new conditions.

The same examples of tested bale images under multiple environmental conditions using the model trained in Step 3 are shown in Fig. 37. These typical results under different conditions show the improvement comparing to Fig. 37.



(a) (b)

(c) (d) (e)

Figure 37 Example of tested bale images under multiple environmental conditions using our framework (YOLOv3+DA): (a) initial condition with good illumination and no shadow in fall; (b) extended condition – early winter with shadow; (c) summer with haze; (d) winter with snow covered; (e) summer with good illumination condition.

4.4.4 Comparison and Advantages

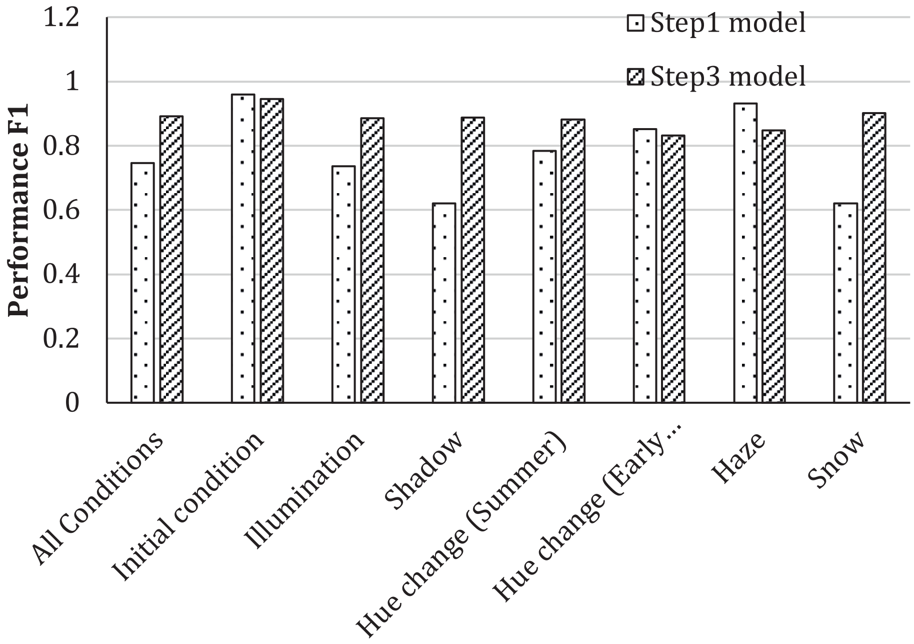


Figure 38 F1 comparison between Step 1 (primary YOLOv3 model) and Step 3 (optimized YOLOv3+DA model) under each condition and mixed all conditions.

To better understand the detecting improvement on images under different environmental conditions, we plot the F1 value between Step 1 and Step 3 under each condition separately. It is obviously discovered that under most conditions, the performance increases substantially except initial condition, Hue change (early winter), and Haze. Because the optimization curve generally slows down after the accuracy is over high value when improving the object detection performance. What we aim is improving the detection accuracy of the conditions with lower accuracy (less than 80%). So in our case, we expect to see the big jump for conditions like, illumination, shadow, Hue change (Summer) and snow which are all less than 75% accuracy. The following results analysis show that we not only keep the original high performance but also increased the performance of some conditions with originally low accuracy.

Firstly, the initial condition already has a high accuracy that over 93% with either model. Similar, the Hue change (early winter) condition also keeps a relevantly high performance around 90% before and after our approach. This method maintains the high accuracy score with a slight change during to the enhancement of training dataset volume and the false negative samples. Meanwhile, this Haze images enlarge the base number when calculating F1 score. Although the haze condition accuracy is high, we can still make improvement by collecting more better-quality images season by season. But this is a long period and continuous collecting work for our lab, that is not the core contribution of this algorithm research Secondly, for conditions like, illumination, shadow, Hue change (Summer) and snow, our method significantly ameliorate the detection accuracy by around 15%, 26%, 10% and 28% respectively. Generally, this YOLOv3+DA model approves its advantages in augmenting detection ability.

Moreover, we estimated the time cost of manually labelling bales in all images as shown in Table 10. Step 1 only need label images under the initial condition with around 90 hours. After that, we have two options to augment the bale detection model. One is to label every new image under all extended conditions with 260 extra hours of work, the other one is to train a CycleGAN model without extra labelling other than the first 90 hours. Since the general F1 score, precision, recall and mAP from the proposed approach are all over 0.9, this is sufficient for this specific task. Thus, the proposed method provides additional advantages of saving time and labor.

Table 8 Manually labelling cost for bales detection with different approaches.

Train Approach	Time Cost (hours)
w/ initial condition images	90
w/ domain adaption images	90
w/ labeled all conditions images[1]	350

[1] "labelled all conditions images" means manually label real images under all conditions then train a model with these labelled data.

4.5 Conclusion

A YOLOv3 bale detection model combining with the Domain Adaptation approach is addressed in this paper, augmenting the ability for crop/bale detection in three seasons, different illumination conditions, and diverse weather conditions. This method is advantageous as it needs limited manually labelling task. In this work, only the images captured under initial condition need to be manually labelled as source-domain data. Then the Domain Adaptation approach, CycleGAN models, are trained to transfer source-domain images to target-domain (images under other conditions) with the same labelled annotation file. We effectively have augmented the training datasets under extended conditions but without extra manual labelling task. After these two steps, we trained the YOLOv3 model again with augmented training datasets. The optimized YOLOv3 model shows a significant improvement in the general detecting performance. This approach decreases the labor and time cost in the way of improving crop quality and yields. Also, it shows strong scalability on many other crops and will significantly reduce the cost of precision agriculture.

Chapter 5 Summary and Future Plan

5.1 Summary

In this thesis, several innovative methods and associated system were created for agriculture navigation, mapping and object detection. These were constructed as three studies, namely field 3D Mesh-mapping, advanced sensors and data fusion, and agriculture objects perceiving. The studies have shown how to logically support the high-level agriculture AI system by (i) Fundamentally creating advanced field mapping and navigation algorithms based on computer vision techniques; (ii) Profitably augmenting the mapping module by a sensing approach with a variety of sensors and data fusion; and (iii) Precisely augmenting agriculture objects perceiving ability for the map using transfer learning and computer vision techniques.

Firstly, a 3D Mesh-mapping system is built assisted by an IoT architecture which is utilized to disperse the computation load and network communication between multiple edge nodes and a single cloud server. The experiment shows reasonable performance in term of accuracy, scalability, data transmission and automation. Next, to improve the Mesh-mapping approach, we present a sensing algorithm, a low-cost, robot-mounted, multidimensional map construction method which gathers landscape information for 3D map construction. This method tracks robot movements, monitors the landscape environment, and links all the factors to the 3D map. The study also provides analytics and data fusion from several specific aspects of the robot working environment. The applications demonstrate the usefulness on task planning and route planning. Finally, diving down to an agriculture object level, an object detection model hybrid with the Domain Adaptation was addressed, augmenting the ability for crop detection in various environmental conditions. The optimized object detection model shows a significant improvement in the general detecting performance.

Based on the experiences with these applications, it is believed all the addressed AI techniques will be useful to the development of future "smart farms." Autonomous and precision agricultural systems mitigate issues of current agriculture. This system is designed to reduce labor issues for the most dangerous and tedious agronomic tasks, improve efficiency, and reduce environmental impacts through better utilization of crop inputs. With the development of computing infrastructure, hardware, and improving algorithms, this system can enable more powerful applications in the future.

5.2 Future Research Directions

The utilization of SLAM and the assumption of the planar Mesh-map are based on the premise that the testing land is planar. Thus, the adoption of SLAM to better serve the navigation problem could still be improved. Other than IMU sensors, more growing status monitoring sensors could be added to detect other context information to boost the performance. These are imagined making SLAM a viable option for automated mapping and navigation systems to enable autonomous agricultural systems. Data association process could be improved in the future, which is implemented in this proposed approach. Although these systems outperform existing methods, they still failed to extract useful features from the images under adverse weather conditions during the data collection process. There it is suggest to test using high accuracy RTK-GPS assistance in the future.

Other applications of these should also be explored. For example, one could apply deep learning techniques to detect obstacles in the field which could potentially cause blockages to farm machines. The ability to detect obstacles allows us to further augment the insight of the mapping method as a 4D or even higher dimension map rather easily. Although this paper targets on reconstructing the 4D map, which helps analyzing crop growing environment, the analysis of crop height point cloud is absent. Future work could be performed to analyze the point clouds with more sophisticated methods.

Planned future work also includes collecting more real images under more specific conditions, generating more synthetic images associate with these conditions, and combining the activate learning method with the CycleGAN model, making the whole pipeline of the algorithm more robust and easier for use. More images could be collected to improve the proposed method, such as real-time images and patterns, image modalities, thermal images, soil maps, and topographic maps. This would lead to “Agriculture-Vision” having a more standardized and inclusive dataset, encouraging more research on improving visual recognition approaches in modern agriculture, including large-scale, multi-channel aerial farmland semantic segmentation.

The methods from this thesis can be applied in other domains other than agriculture. For example, our system overcomes the limitations of existing IMU-based solutions. Combining with the deep learning and object detection algorithms, we can provide fruitful contextual information for driver behavior profiling in transportation domain. The future work will leverage mobile sensing and computer vision techniques to extract various context information of the driver and the surrounding environments. Further driving behaviour evaluation could be conducted using the extracted information. Base on this work, one could build a low-cost embedded computer with deep learning inference accelerator. The prototype can be deployed to a regular vehicle and tested in real-world environments. This system can process data in real time and provide a good understanding of each driving behaviour. It is believed such a real-time sensing and analysis system can enable a wide range of applications.

References

[1]. USDA National Agricultural Statistics Service, 2017 Census of Agriculture. Complete data available at www.nass.usda.gov/AgCensus.

[2]. U.S. Census Bureau (2020). U.S. and World Population Clock. Retrieved from https://www.census.gov/popclock/

[3]. Godfray, H., et al., "The future of the global food system," *Philosophical transactions of the Royal Society of London. Series B, Biological sciences* 365.1554(2010): 2769-2777.

[4]. Massey, R.E., D.B. Myers, N.R. Kitchen, and K.A. Sudduth. 2008. "Profitability Maps as an Input for Site-Specific Management Decision Making," *Agronomy Journal* 100(1):52-59.

[5]. Van Raij, B., H. Cantarella, and J. A. Quaggio. 2002. "Rationale of the Economy of Soil Testing," *Communications in Soil Science and Plant Analysis* 33(15-18):2521-36.

[6]. Schimmelpfennig, D. (2016). *Farm profits and adoption of precision agriculture* (No. 1477-2016-121190).

[7]. Van Evert F, van der Voet P, Van Valkengoed E, Kooistra L, Kempenaar C (2012) Satellite-based herbicide rate recommendation for potato haulm killing. *Eur J Agron* 43:49–57

[8]. Basso B, Ritchie JT, Cammarano D, Sartori L (2011) A strategic and tactical management approach to select optimal N fertilizer rates for wheat in a spatially variable field. *Eur J Agron* 35:215–222

[9]. Basso B, Sartori L, Cammarano D, Grace PR, Fountas S, Sorensen C (2012) Environmental and economic evaluation of N fertilizer rates 1 in a maize crop in Italy: a spatial and temporal analysis. *Biosyst Eng* 113:103–111

[10]. Pedersen SM (2003) Precision farming – technology assessment of site-specific input application in cereals. *Department of Manufacturing Engineering and Management, Technical University of Denmark. Ph.D dissertation*, 343 p

[11]. Tian, H., Wang, T., Liu, Y., et al. "Computer vision technology in agricultural automation—A review," *Information Processing in Agriculture* 7.1 (2020):1-19.

[12]. Romualdo, L. M., Luz, P. H. C., Devechio, F. F. S., et al. "Use of artificial vision techniques for diagnostic of nitrogen nutritional status in maize plants," *Computers and electronics in agriculture* 104 (2014): 63-70.

[13]. Pérez-Zavala, R., Torres-Torriti, M., Cheein, F. A., et al. "A pattern recognition strategy for visual grape bunch detection in vineyards," *Computers and Electronics in Agriculture* 151 (2018): 136-149.

[14]. Mahajan, S., Das, A., Sardana, H. K., "Image acquisition techniques for assessment of legume quality," *Trends in Food Science & Technology*, 42.2 (2015), pp. 116-133.

[15]. Barbedo, J.G.A., "A review on the main challenges in automatic plant disease identification based on visible range images," *Biosystems Engineering* 144 (2016): 52-60.

[16]. Story, D., Kacira, M., "Design and implementation of a computer vision-guided greenhouse crop diagnostics system," *Machine vision and applications* (2015): 495-506.

[17]. Patrício, D. I., Rieder, R., "Computer vision and artificial intelligence in precision agriculture for grain crops: A systematic review*," Computers and electronics in agriculture* 153 (2018): 69-81.

[18]. Tillett, N. D., Hague, T., Grundy, A. C., et al. "Mechanical within-row weed control for transplanted crops using computer vision," *Biosystems Engineering* 99.2 (2008):171-178.

[19]. Mohanty, S. P., Hughes, D. P., Salathé, M., "Using deep learning for image-based plant disease detection," *Frontiers in plant science* 7 (2016): 1419.

[20]. Rastogi, A., Arora, R., Sharma, S., "Leaf disease detection and grading using computer vision technology & fuzzy logic*," signal processing and integrated networks (SPIN)* (2015).

[21]. Voivontas, D., Assimacopoulos, D., Koukios, E. G., "Aessessment of biomass potential for power production: a GIS based method," *Biomass and bioenergy* 20.2 (2001): 101-112.

[22]. Sokhansanj, S., Kumar, A., Turhollow, A. F., "Development and implementation of integrated biomass supply analysis and logistics model (IBSAL)," *Biomass and Bioenergy* 30.10 (2006): 838-847.

[23]. Habib, A., Han, Y., Xiong, W., He, F., Zhang, Z., & Crawford, M. (2016). Automated ortho-rectification of UAV-based hyperspectral data over an agricultural field using frame RGB imagery. *Remote Sensing*, 8(10), 796.

[24]. Ward, S., L., Hensler, J., Alsalam, B. H. Y.; et al. “Autonomous UAVs Wildlife Monitoring and Tracking Using Thermal Imaging and Computer vision,” *IEEE Aerospace conferece (2016)*.

[25]. Xiang, H., Tian, L., “Method for automatic georeferencing aerial remote sensing (RS) images from an unmanned aerial vehicle (UAV) platform,” *Biosystems Engineering* 108 (2011): 104-113.

[26]. Hunt, E. R., Cavigelli, M., Daughtry, C. S., et al. “Evaluation of digital photography from model aircraft for remote sensing of crop biomass and nitrogen status,” *Precision Agriculture* 6 (2005): 359-378.

[27]. Rieke, M., Foerster, T., Geipel, J., et al. “High-Precision Positioning And Real-Time Data Processing Of Uav-Systems,” *International Archives of Photogrammetry Remote Sensing and Spatial Information Sciences* 38 (2011): 1-C22.

[28]. Oberti, R., Marchi, M., Tirelli, P., et al. “Automatic detection of powdery mildew on grapevine leaves by image analysis: Optimal view-angle range to increase the sensitivity,” *Computers and Electronics in Agriculture* 104 (2014): 1-8.

[29]. Pourreza, A., Lee, W. S., Ehsani, R., et al. “An optimum method for real-time in-field detection of Huanglongbing disease using a vision sensor,” *Computers and Electronics in Agriculture* 110 (2015): 221-232.

[30]. Maharlooei, M., Sivarajan, S., Bajwa, S. G., et al. “Detection of soybean aphids in a greenhouse using an image processing technique,” *Computers and electronics in agriculture* 132 (2017): 63-70.

[31]. Toseef, M., Khan, M. J., “An intelligent mobile application for diagnosis of crop diseases in Pakistan using fuzzy inference system,” *Computers and Electronics in Agriculture* 153 (2018): 1-11.

[32]. Barnea, E., Mairon, R., Ben-Shahar, O., “Colour-agnostic shape-based 3D fruit detection for crop harvesting robots,” *Biosystems Engineering* 146 (2016): 57-70.

[33]. Lehnert, C., English, A., McCool, C., et al. “Autonomous sweet pepper harvesting for protected cropping systems,” *IEEE Robotics and Automation Letters* 2.2 (2017): 872-879.

[34]. The International Society of Precision Agriculture Retrieved from https://www.ispag.org/

[35]. Hussnain, M., Sharjeel, M., Chaudhry, S. R., Hussain, S. A., Raza, I., and Mirza, J. S., “Investigating multi-topological zigbee based wireless sensor network in precision agriculture,” Journal of Basics and Applied Scientific Research (2013).

[36]. Mareels, I., Weyer, E., Ooi, S. K., Cantoni, M., Li, Y., and Nair, G., “Systems engineering for irrigation systems: Successes and challenges,” IFAC Proceedings Volumes 38.1 (2005): 1-16.

[37]. Rembold, F., et al., “Using low resolution satellite imagery for yield prediction and yield anomaly detection,” *Remote Sensing* 5.4 (2013): 1704–1733.

[38]. Zainuddin, K., et al., “Verification test on ability to use low-cost UAV for quantifying tree height,” *2016 IEEE 12th International Colloquium on Signal Processing & Its Applications* (2016).

[39]. Sa, I., et al., “Deepfruits: A fruit detection system using deep neural networks,” *Sensors* 16. 8 (2016): 1224-8220.

[40]. Furukawa, Y., and Ponce, J., “Accurate, dense, and robust multi-view stereopsis,” *IEEE Trans. Pattern Anal. Machine Intell* 32.8 (2010): 1362–1376.

[41]. Griffith, S., and Pradalier, C., “Reprojection flow for image registration across seasons,” *27th British Machine Vision Conference* (2016).

[42]. Naseer, T., Suger, B., Ruhnke, M., and Burgard, W., “Vision-based markov localization across large perceptual changes,” *European Conf. on Mobile Robots* (2015).

[43]. Lowry, S. et al., “Visual place recognition: A survey,” *IEEE Trans. Robotics* 3.1 (2016): 1–19.

[44]. Beall, C., and Dellaert, F., “Appearance-based localization across seasons in a metric map,” *IROS Workshop on Planning, Perception and Navigation for Intelligent Vehicle* (2014).

[45]. Stein, M., Bargoti, S., and Underwood, J., “Image based mango fruit detection, localisation and yield estimation using multiple view geometry,” *Sensors* 16.11 (2016):1424-8220.

[46]. Sundmaeker, H., Verdouw, C., and Wolfert, S., “Digitising the Industry-Internet of Things connecting physical, digital and virtual worlds,” *Internet of food and farm* (2016).

[47]. Kaloxylos, A., Groumas, A., and Sarris, V., “A cloud-based Farm Management System: Architecture and implementation,” *Computers and Electronics in Agriculture* 100 (2014): 168-179.

[48]. Tse-Chuan, H., Yang, H., Chung, Y., and Hsu, C., “A Creative IoT agriculture platform for cloud fog computing,” *Sustainable Computing: Informatics and Systems* (2018):2210-5379.

[49]. Christopher, B., et al., “IoT in agriculture: Designing a Europe-wide large-scale pilot,” *IEEE communications magazine* 55. 9 (2017): 26-33.

[50]. Kerry, T., et al., “Farming the web of things,” *IEEE Intelligent Systems* 8.6 (2016):12-19.

[51]. Muangprathub, J., Boonnam, N., Kajornkasirat, S., Lekbangpong, N., Wanichsombat, A., & Nillaor, P. (2019). IoT and agriculture data analysis for smart farm. *Computers and electronics in agriculture*, 156, 467-474.

[52]. Jirapond, M., et al., “IoT and agriculture data analysis for smart farm,” *Computers and electronics in agriculture* 156 (2019): 467-474.

[53]. Mohit, T., Byabazaire, J., Jalodia, N., Davy, A., Olariu, C., and Malone, P., “Machine learning based fog computing assisted data-driven approach for early lameness detection in dairy cattle,” *Computers and Electronics in Agriculture* 171 (2020): 0168-1699.

[54]. Mohit, T., et al., “SmartHerd management: A microservices-based fog computing–assisted IoT platform towards data-driven smart dairy farming,” *Software: Practice and Experience* 49.7 (2019): 1055-1078.

[55]. Tim, W., et al., “Sensor and actuator networks: Protecting environmentally sensitive areas,” *IEEE Pervasive Computing* 8.1(2008): 30-36.

[56]. Nandurkar, S. R., Thool, V. R., and Thool, R. C., “Design and development of precision agriculture system using wireless sensor network,” *Automation, Control, Energy and Systems (ACES)* (2014).

[57]. Anderson, N. W., “Method and system for identifying an edge of a plant,” *U.S. Patent* (2011)

[58]. Rouveure, R., Nielsen, M., Petersen, A., et al. “The QUAD-AV Project: Multi-sensory approach for obstacle detection in agricultural autonomous robotics,” *Agricultural Engineering CIGR-Ageng* (2012).

[59]. Zhao, W., Xu, L., Bai, J., Ji, M., & Runge, T., “Sensor-based risk perception ability network design for drivers in snow and ice environmental freeway: a deep learning and rough sets approach,” *Soft computing* 22.5 (2018): 1457-1466.

[60]. Beall, C.; Dellaert, F.; Appearance-based localization across seasons in a metric map. *IROS Workshop on Planning, Perception and Navigation for Intelligent Vehicles (PPNIV)*, 2014.

[61]. Hergt, P., Krieger, P., “Paper 10: radial forces in centrifugal pumps with guide vanes,” *Institution of Mechanical Engineers* (1969).

[62]. Abrams, R. A., Christ, S. E., “Motion onset captures attention,” *Psychological Science* 14.5 (2003): 427-432.

[63]. Estes, R. A., Bynum, J. R., Riggs, R., et al. “Gyroscopic steering tool using only a two-axis rate gyroscope and deriving the missing third axis”: U.S. *Patent* (2007).

[64]. Lane, N. D., Miluzzu, E., Lu, H., Peebles, D., Choudury, T., Campbell, A. T., “A survey of mobile phone sensing,” *IEEE Communications magazine* 48.9 (2010): 140-150.

[65]. Herrera, J. C., Work, D. B., Herring, R., Ban, X. G., Jacobson, Q., Bayen, A. M., “Evaluation of traffic data obtained via GPS-enabled mobile phones: The mobile century, field experiment,” *Transportation Research Part C-Emerging Technologies* 18.4 (2009): 568-583.

[66]. Dai, J. P., Teng, J., Bai, X. L., Shen, Z. H., Xuan, D., “Mobile phone based drunk driving detection,” *Pervasive Computing Technologies for Healthcare* (2010): 1-8.

[67]. Mantouka, E. G., Barmpounakis, E. N., Vlahogianni, E. I., “Mobile sensing and machine learning for identifying driving safety profiles,” *Transportation Research Board 97th Annual Meeting* (2018).

[68]. Papadimitriou, E., Argyropoulou, A., Tselentis, D. I., Yannis, G., “Analysis of driver behavior through smartphone data: the case of mobile phone use while driving,” *Transportation Research Board 97th Annual Meeting* (2019).

[69]. Kanarachos, S., Christopoulos, S. R. G., Chroneos, A., “Smartphones as an integrated platform for monitoring driver behaviour: The role of sensor fusion and connectivity,” *Transportation Research Part C: Emerging Technologies* (2018).

[70]. Kang, L., Banerjee S., “Practical driving analytics with smartphone sensors,” *2017 IEEE Vehicular Networking Conference* (2017).

[71]. Fazeen, M., Gozick, B., Dantu, R., Bhukhiya, M., González, M. C., “Safe driving using mobile phones,” *IEEE Transactions on Intelligent Transportation Systems* 13.3 (2012): 1462-1468.

[72]. Bruwer, F. J., Booysen, M. J., “Comparison of GPS and MEMS support for smartphone-based driver behavior monitoring,” *IEEE Symposium Series on Computational Intelligence* (2015): 434-441.

[73]. Zhao, W., Xu, L., Bai, J., Ji, M., Runge, T., “Sensor-based risk perception ability network design for drivers in snow and ice environmental freeway: A deep learning and rough sets approach,” *Soft computing* 22.5 (2018) 1457-1466.

[74]. Gunawan, F. E., Yanfi, Soewito, B. A., “vibratory-based method for road damage classification,” *International Seminar on Intelligent Technology & Its Applications* (2015): 1-4.

[75]. Dang-Nhac, L., Duc-Nhan N., Thi-Hau N., Ha-Nam, N., “Vehicle mode and driving activity detection based on analyzing sensor data of smartphones,” *Sensors* 18.4 (2018): 036-.

[76]. Krizhevsky, A., Sutskever, I., Hinton. G. E., “Imagenet classification with deep convolutional neural networks,” *25th International Conference on Neural Information Processing Systems* (2012).

[77]. Richter, S. R., Vineet, V., Roth, S., et al. “Playing for data: Ground truth from computer games,” *European Conference on Computer Vision* (2016).

[78]. Othman, E., Bazi, Y., Melgani, F, et al. Domain adaptation network for cross-scene classification. *IEEE Transactions on Geoscience and Remote Sensing* 55.8 (2017): 4441-4456.

[79]. Li, X., Ye, M., Fu, M., et al. “Domain adaption of vehicle detector based on convolutional neural networks,” *International Journal of Control, Automation and Systems* 13.4 (2015): 1020-1031.

[80]. Song, S., Yu, H., Miao, Z., et al. “Domain Adaptation for Convolutional Neural Networks-Based Remote Sensing Scene Classification,” *IEEE Geoscience and Remote Sensing Letters* 16.8 (2019): 1324-1328.

[81]. Wikipedia. [n. d.]. https://en.wikipedia.org/wiki/Autonomous car. ([n. d.]).

[82]. Todd, L., “Autonomous vehicle implementation predictions,” Report (2014).

[83]. AutoX. “Camera-first AI brings self-driving cars out of the lab and into the real world,” https://www.autox.ai/. (2016).

[84]. Autonomous Mercedes-Benz Research & Development. [n. d.]. http://mbrdna.com/divisions/autonomous-driving/. ([n. d.]).

[85]. Waymo. On the road to fully self-driving, Waymo safety report. https://storage.googleapis.com/sdc-prod/v1/safety-report/waymo-safety-report-2017-10.pdf. (2016).

[86]. Daily, M., [n. d.]. http://www.dailymail.co.uk/sciencetech/article-4889594/Google-s-spent-1-1-BILLION-self-driving-cars. html. ([n. d.]).

[87]. Chen, X., Ma, H., Wan, J., Li, B., Xia, T., “Multi-View 3D Object Detection Network for Autonomous Driving,” *IEEE CVPR* (2017).

[88]. Leonard, J., Jonathan H., Seth T., et al. “A perception-driven autonomous urban vehicle,” *Journal of Field Robotics* (2008).

[89]. Eun-Kyu, L., Mario, G., Giovanni, P., Uichin, L., JaeHan, L., “Internet of Vehicles: From intelligent grid to autonomous cars and vehicular fogs,” *International Journal of Distributed Sensor Networks* (2016).

[90]. Ankit S., Balakrishnan, C., Godfrey, P., Bruce M., “The internet at the speed of light,” *In HotNets*. ACM (2014).

[91]. Mamta, A., Abhishek, R., Navrati, S., “Next generation 5G wireless networks: A comprehensive survey,” *IEEE Communications Surveys & Tutorials* (2016).

[92]. Zhang, S., Xu, X., Wu, Y., Lu, L., “5G: Towards energy-efficient, low-latency and high-reliable communications networks,” *In ICCS. IEEE* (2014).

[93]. Stonebraker, M., Cetintemel, U., Zdonik, Stan., “The 8 requirements of real-time stream processing,” *ACM SIGMOD* (2005).

[94]. Song, Z., Shangguan, L., Jamieson, K., “Wi-Fi Goes to Town: Rapid Picocell Switching for WirelessTransit Networks,” *In ACM SIGCOMM Computer Communication Review* (2017).

[95]. Kyungmin, L., David, C., Eduardo, C., et al. “Outatime: Using speculation to enable low-latency continuous interaction for mobile cloud gaming,” *In MobiSys. ACM.* (2015).

[96]. Huang, J., Qian, F., Guo, Y., Zhou, Y., Xu, Q., Mao, Z., Sen, S., Spatscheck, O., “An in-depth study of LTE: effect of network protocol and application behavior on performance,” *In ACM SIGCOMM. ACM.* (2013).

[97]. Najah, A. A., Abd-Elhamid, M. T., Hossam, S. H., “Quality of service in 3GPP R12 LTE-advanced,” *IEEE Communications Magazine* (2013).

[98]. Tu, G., Li, C., Peng, C., Yuan, Z., Li, Y., Zhao, X., Lu. S., “VoLTE*: A Lightweight Voice Solution to 4G LTE Networks,” *HotMobile*. ACM. (2016).

[99]. Lane, N. D., Miluzzu, E., Lu, H., Peebles, D., Choudury, T., Campbell, A. T., “A survey of mobile phone sensing,” *IEEE Communications magazine* 48.9 (2010): 140-150.

[100]. Herrera, J. C., Work, D. B., Herring, R., Ban, X. G., Jacobson, Q., Bayen, A. M., “Evaluation of traffic data obtained via GPS-enabled mobile phones: The mobile century, field experiment,” *Transportation Research Part C-Emerging Technologies* 18.4 (2009): 568-583.

[101]. Dai, J. P., Teng, J., Bai, X. L., Shen, Z. H., Xuan, D., "Mobile phone based drunk driving detection," *Pervasive Computing Technologies for Healthcare* (2010): 1-8.

[102]. Mantouka, E. G., Barmpounakis, E. N., Vlahogianni, E. I., "Mobile sensing and machine learning for identifying driving safety profiles," *Transportation Research Board 97th Annual Meeting* (2018).

[103]. Papadimitriou, E., Argyropoulou, A., Tselentis, D. I., Yannis, G., "Analysis of driver behavior through smartphone data: the case of mobile phone use while driving," *Transportation Research Board 97th Annual Meeting* (2019).

[104]. Kanarachos, S., Christopoulos, S. R. G., Chroneos, A., "Smartphones as an integrated platform for monitoring driver behaviour: The role of sensor fusion and connectivity," *Transportation Research Part C: Emerging Technologies* (2018).

[105]. Kang, L., Banerjee S., "Practical driving analytics with smartphone sensors," *2017 IEEE Vehicular Networking Conference* (2017).

[106]. Fazeen, M., Gozick, B., Dantu, R., Bhukhiya, M., González, M. C., "Safe driving using mobile phones," *IEEE Transactions on Intelligent Transportation Systems* 13.3 (2012): 1462-1468.

[107]. Bruwer, F. J., Booysen, M. J., "Comparison of GPS and MEMS support for smartphone-based driver behavior monitoring," *IEEE Symposium Series on Computational Intelligence* (2015): 434-441.

[108]. Zhao, W., Xu, L., Bai, J., Ji, M., Runge, T., "Sensor-based risk perception ability network design for drivers in snow and ice environmental freeway: A deep learning and rough sets approach," *Soft computing* 22.5 (2018) 1457-1466.

[109]. Gunawan, F. E., Yanfi, Soewito, B. A., "vibratory-based method for road damage classification," *International Seminar on Intelligent Technology & Its Applications* (2015): 1-4.

[110]. Hao,P., Ban,X.G., Yu,J.W., Kinematic equation based vehicle queue location estimation method for signalized intersections using mobile sensor data," *J. Intell. Transp. Syst.* 19 (2015): 256–272.

[111]. Zhu, Y., Comaniciu, D., Pellkofer, M., Koehler, T., Reliable detection of overtaking vehicles using robust information fusion," *IEEE Trans. Intell. Transp. Syst.* 7 (2007): 401–414.

[112]. Bottero, M., Chiara, B.D., Deflorio, F. P., "Wireless sensor networks for traffic monitoring in a logistic centre," *Transp. Res. Part C Emerg. Technol.* 26 (2014): 99–124.

[113]. Lefèvre, S., Vasquez, D., Laugier, C., “A survey on motion prediction and risk assessment for intelligent vehicles,” *ROBOMECH J.* 1 (2014): 1–14.

[114]. Kassem, N., Kosba, A. E., Youssef, M., “RF-based vehicle detection and speed estimation,” *In Proceedings of the IEEE Vehicular Technology Conference* (2012).

[115]. Won, S. H. P., Golnaraghi, F., Melek, W. W., “A Fastening Tool Tracking System Using an IMU and a Position Sensor With Kalman Filters and a Fuzzy Expert System,” *IEEE Transactions on Industrial Electronics* 56.5 (2009):1782-1792.

[116]. Feng, T., Liu Z., Kwon, K. A., et al. “Continuous mobile authentication using touchscreen gestures,” *Homeland Security (HST)* (2012): 451-456.

[117]. Chao, H., Mao. L., Shuang. S., et al. “A Cubic 3-Axis Magnetic Sensor Array for Wirelessly Tracking Magnet Position and Orientation,” *IEEE Sensors Journal* 10.5 (2010): 903-913.

[118]. “Video clips,” *BMJ* (2014).

[119]. Mayagoitia, R. E., Nene, A. V., “Veltink P H. Accelerometer and rate gyroscope measurement of kinematics: an inexpensive alternative to optical motion analysis systems,” *Journal of Biomechanics* 35.4 (2002): 537-542.

[120]. Bouten, C. V., Koekkoek, K. T., Verduin, M., et al. “A triaxial accelerometer and portable data processing unit for the assessment of daily physical activity,” *IEEE Transactions on Biomedical Engineering* 44.3 (1997): 136-47.

[121]. Du, X., Elkhamy, M., Lee, J., et al. “Fused DNN: A deep neural network fusion approach to fast and robust pedestrian detection,” 2016.

[122]. Zhao, W., Xu, L., Dong, Z. S., Qi, B., Qin, L., “Improving transfer feasibility for older travelers inside high-speed train station,” *Transportation Research Part A: Policy and Practice* 113 (2018): 302-317.

[123]. Pritsker, A. A. B., “Introduction to stimulation and Slam II,” 1986.

[124]. Reutebuch, S. E., Andersen, H. E., and McGaughey, R. J., “Light detection and ranging (LIDAR): an emerging tool for multiple resource inventory,” Journal of Forestry 103.6 (2015): 286-292.

[125]. Davison, A. J., Reid, I. D., Molton, N. D., “MonoSLAM: Real-time single camera SLAM,” *IEEE Transactions on Pattern Analysis & Machine Intelligence* 6 (2007): 1052-1067.

[126]. Georg, K., and David, M., “Parallel Tracking and Mapping for Small AR Workspaces,” *6th IEEE and ACM International Symposium on Mixed and Augmented Reality* (2007).

[127]. Mur-Artal, R., Montiel, J. M. M., and Tardos, J. D., “ORB-SLAM: a versatile and accurate monocular SLAM system,” *IEEE Transactions on Robotics* 31.5 (2015): 1147-1163.

[128]. Newcombe, R. A., Lovegrove, S. J. and Davison, A. J. “DTAM: Dense tracking and mapping in real-time,” *IEEE International Conference on Computer Vision* (2011).

[129]. Engel, J., Schöps, T. and Cremers, D., “LSD-SLAM: Large-scale direct monocular SLAM,” *European Conference on Computer Vision* (2014).

[130]. Shi, W., Cao, J., and Zhang, Q., “Edge computing: Vision and challenges,” *IEEE Internet of Things Journal* 3.5 (2016): 637-646.

[131]. Taleb, T., Dutta, S., and Ksentini, A., “Mobile edge computing potential in making cities smarter,” *IEEE Communications Magazine* 55. 3 (2017): 38-43.

[132]. Liu, P., Willis, D., and Banerjee, S., “ParaDrop: Enabling Lightweight Multi-tenancy at the Network's Extreme Edge Edge Computing,” *2016 IEEE/ACM Symposium on Edge Computing (SEC)* (2016).

[133]. Pang, H. H., and Tan, K. L., “Authenticating query results in edge computing,” *20th International Conference on Data Engineering* (2004).

[134]. Wang, S., Zhang, X., and Zhang, Y., “A survey on mobile edge networks: Convergence of computing, caching and communications,” *IEEE Access* 5 (2017): 6757-6779.

[135]. Kumar, N., Zeadally, S., and Rodrigues, J. J. P. C., “Vehicular delay-tolerant networks for smart grid data management using mobile edge computing,” *IEEE Communications Magazine* 54.10 (2016): 60-66.

[136]. Huletski, A., Kartashov, D., and Krinkin, K., “Evaluation of the modern visual SLAM methods,” *2015 Artificial Intelligence and Natural Language and Information Extraction, Social Media and Web Search FRUCT Conference* (2015).

[137]. Klein, G., and Murray, D., “Improving the agility of keyframe-based SLAM,” *European Conference on Computer Vision* (2008).

[138]. Kang, L., Zhao, W., Qi, B., and Banerjee, S., “Augmenting self-driving with remote control: Challenges and directions,” *19th International Workshop on Mobile Computing Systems & Applications*.

[139]. Qi, B., et al., "DrivAid: Augmenting driving analytics with multi-modal information," *IEEE Vehicular Networking Conference* (2018).

[140]. Zhao W., et al., "Vivid: Augmenting Vision-Based Indoor Navigation System with Edge Computing," *IEEE Access* 8 (2020): 42909-42923.

[141]. Zhao, W., Yin, J. T., Wang, X. H., et al. "Real-Time Vehicle Motion Detection and Motion Altering for Connected Vehicle: Algorithm Design and Practical Applications," *Sensors (Basel)* 19.19 (2019): 4108.

[142]. Ramey, R., "Boost serialization library," URL: www.boost.org/doc/libs/release/libs/serialization, (2008).

[143]. Zhu, W., Luo, C., Wang, J., & Li, S. (2011). Multimedia cloud computing. *IEEE Signal Processing Magazine*, *28*(3), 59-69.

[144]. Voisin-Denoual, M., "Monocular Visual Odometry for Underwater Navigation: An examination of the performance of two methods," 2018.

[145]. Yin, W., LUO, Y., and LI, S., "Camera calibration based on OpenCV," *Computer Engineering and Design* (2007): 1-063.

[146]. Wang X., et al., "Temporal Frame Sub-Sampling for Video Object Tracking," *Journal of Signal Processing Systems* (2019): 1-13.

[147]. Myalo, O. V., Myalo, V. V., Prokopov, S. P., Solomkin, A. P., and Soynov, A. S., "Theoretical substantiation of machine-tractor fleet technical maintenance system on the example of Omsk region agricultural enterprises," *In Series: Journal of Physics: Conf. Series* 1059 (2018): 012005).

[148]. Nadkarni, P. M., "Management of evolving map data: data structures and algorithms based on the framework map," *Genomics* 30.3 (1995): 565.

[149]. Li, L., Yao, J., Xie, R., et al. "Laser-Based Slam with Efficient Occupancy Likelihood Map Learning for Dynamic Indoor Scenes," *Isprs Annals of Photogrammetry Remote Sensing & Spatial Informa* 3.4 (2016): 119-126.

[150]. Chen, F., Wang, J. M., Sun, B. G., et al. "Relationships of plant species distribution in different strata of Pinus yunnanensis forest with landform and climatic factors," *Chinese Journal of Ecology* (2012).

[151]. Frolking, S., Qiu, J., Boles, S., Xiao, X., Liu, J., Zhuang, Y., ... and Qin, X., "Combining remote sensing and ground census data to develop new maps of the distribution of rice agriculture in China," *Global Biogeochemical Cycles* 16.4 (2002): 38-1.

[152]. Hall, A., Lamb, D. W., Holzapfel, B., and Louis, J., "Optical remote sensing applications in viticulture-a review," *Australian journal of grape and wine research* 8.1 (2002): 36-47.

[153]. Zhao, W., Wang, W., Wang, X., Jin, Z., Li, Y., adn Runge, T., "A low-cost simultaneous localization and mapping algorithm for last-mile delivery," *ICTIS. IEEE* (2019).

[154]. Valin, H., Sands, R. D., Vander Mensbrugghe, D., et al. "The future of food demand: understanding differences in global economic models," *Agricultural Economics* 45.1 (2014): 51-67.

[155]. Cheng, F. C., Huang, S. C., Ruan, S. J., "Illumination-sensitive background modeling approach for accurate moving object detection," *IEEE Transactions on broadcasting* 57.4 (2011): 794-801.

[156]. Zhao, W., Xu, L. J., Xi, S. X., et al. "A Sensor-Based Visual Effect Evaluation of Chevron Alignment Signs' Colors on Drivers through the Curves in Snow and Ice Environment," *Journal of Sensors* (2017): 9168525.

[157]. Hornberg, A., "Ed. Handbook of machine and computer vision: the guide for developers and users," *Wiley-VCH: Weinheim* (2017).

[158]. BAWEJA, H. S., Parhar, T., Nuske, S., "Early-season vineyard shoot and leaf estimation using computer vision techniques," *2017 ASABE Annual International Meeting* 2017.

[159]. Lin, Y., Chen, J., Cao, Y., et al. "Cross-domain recognition by identifying joint subspaces of source domain and target domain," *IEEE Transactions on Cybernetics* 47.4 (2016): 1090-1101.

[160]. Ren, S., He, K., Girshick, R., et al. "Faster R-CNN: Towards real-time object detection with region proposal networks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*39.6 (2015): 91-99.

[161]. Redmon, J., Divvala, S., Girshick, R., et al. "You only look once: Unified, real-time object detection," *Computer Vision and Pattern Recognition (CVPR)* (2016).

[162]. Mu, K., Hui, F., Zhao, X., "Multiple Vehicle Detection and Tracking in Highway Traffic Surveillance Video Based on SIFT Feature Matching," *Journal of Information Processing Systems* 12.2 (2016): 183-195.

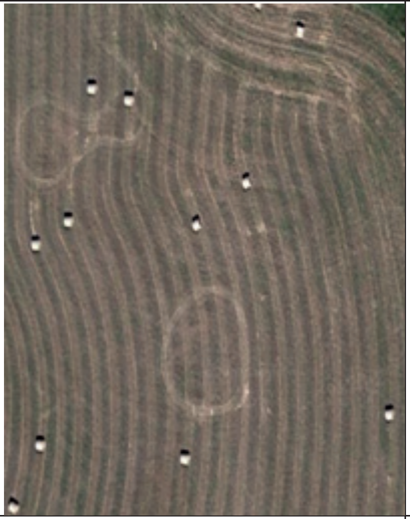
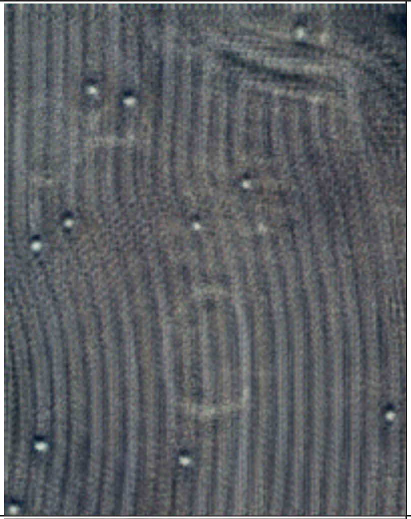
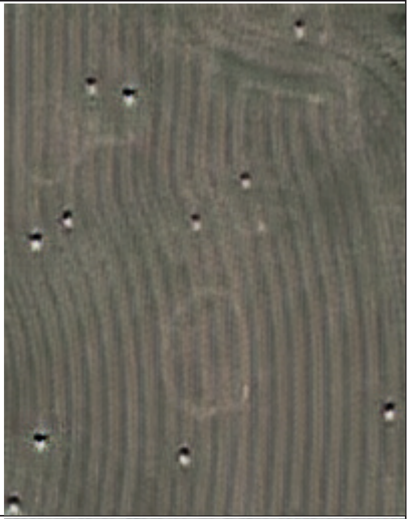
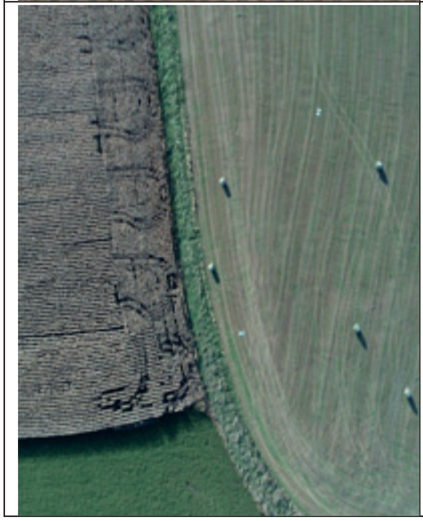
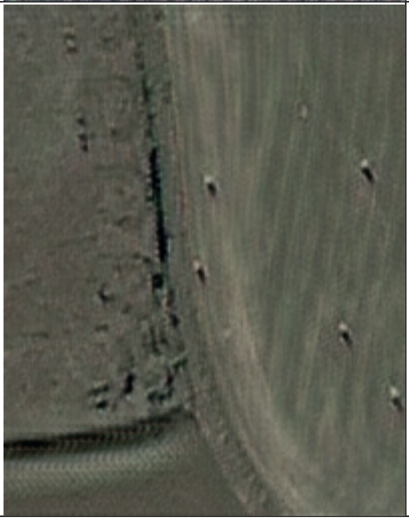
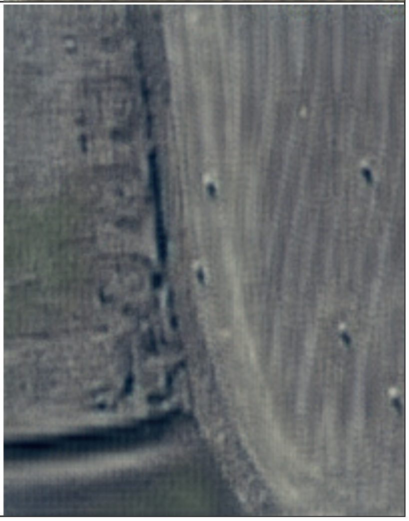
[163]. Zhu, J. Y., Park, T., Isola P., et al. "Unpaired image-to-image translation using cycle-consistent adversarial networks," *2017 IEEE International Conference on Computer Vision* (2017).

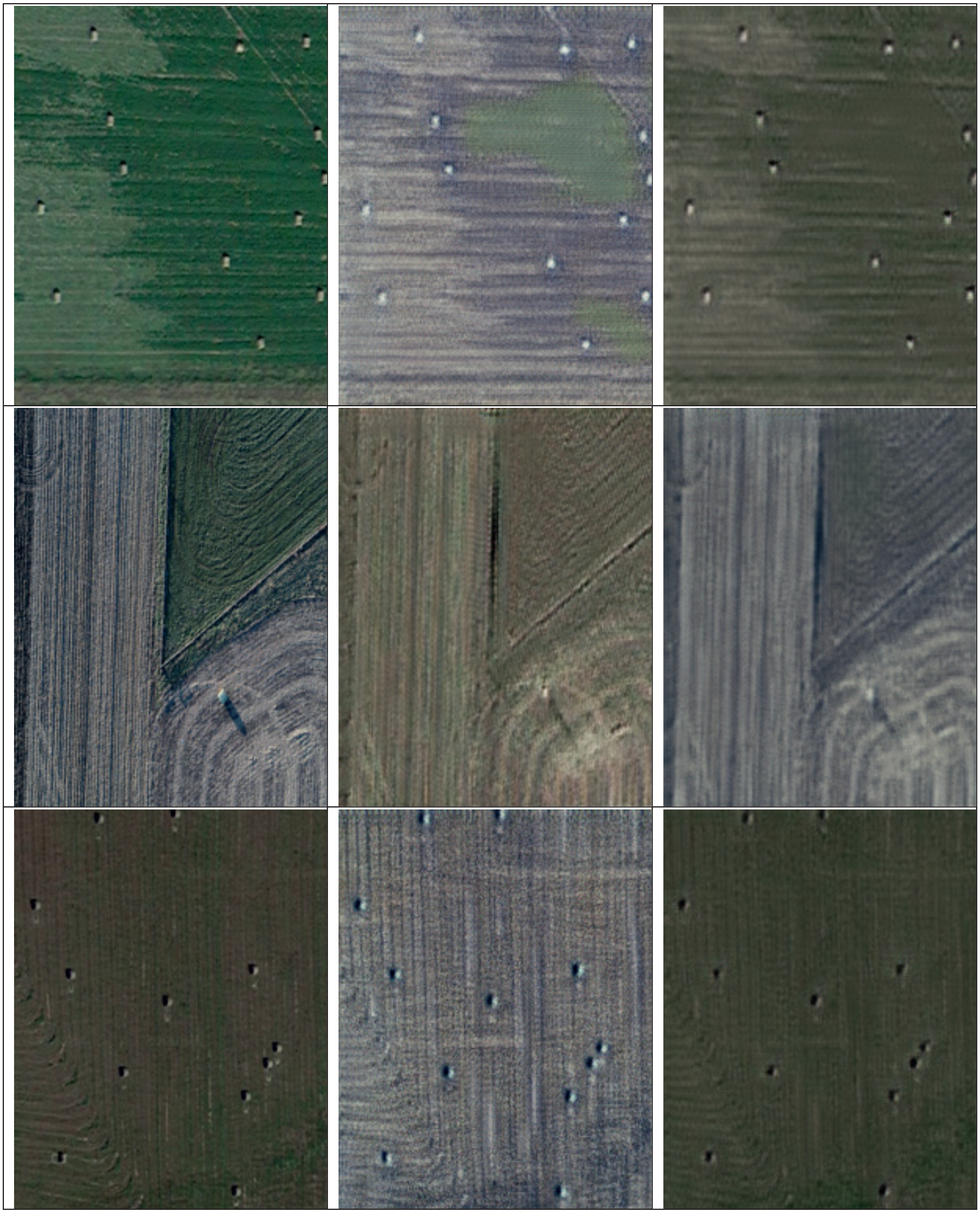
[164]. Tzutalin, Labelimg, Available online: https://github.com/tzutalin/labelImg, 2015.

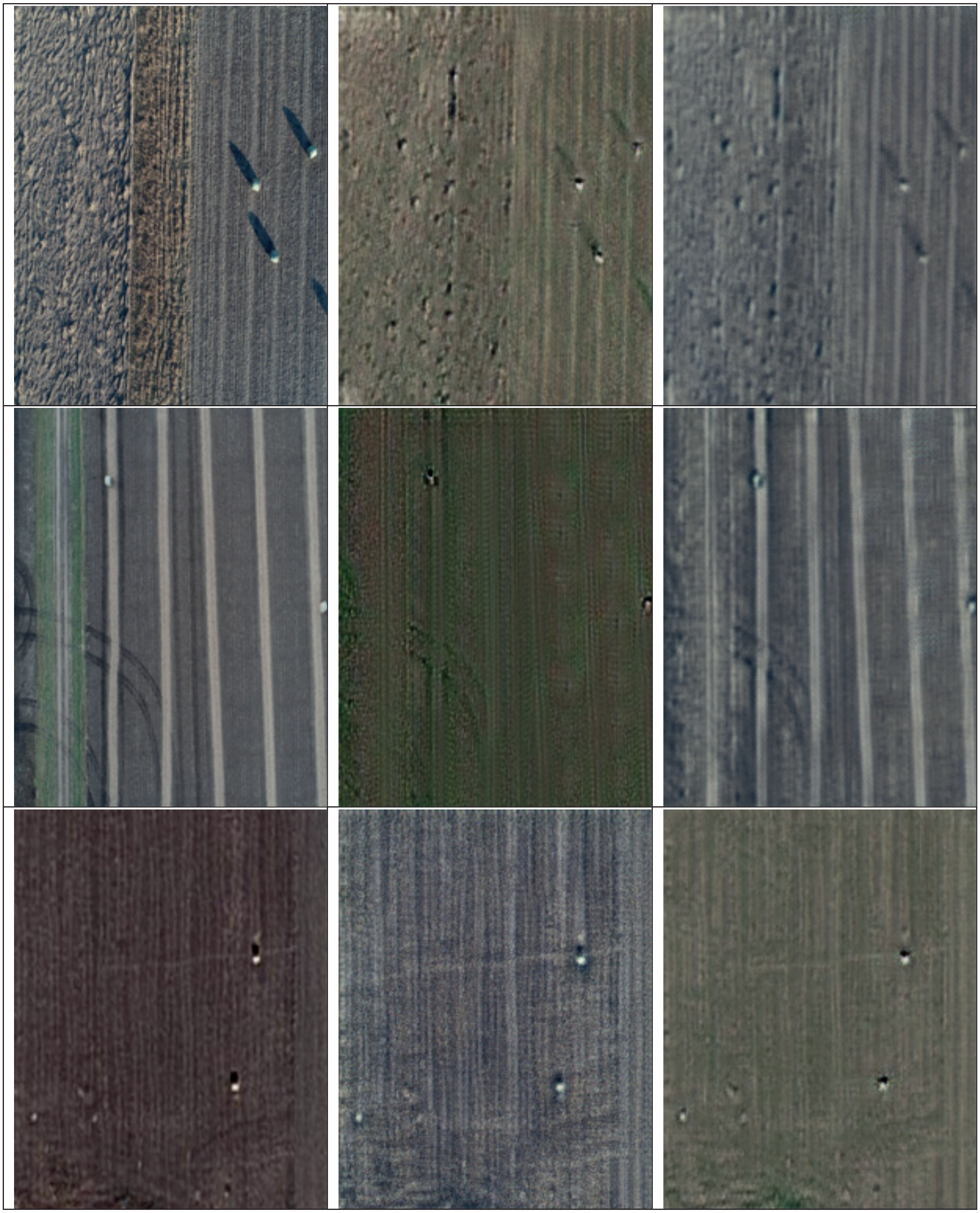
[165]. Kentaro Wada, labelme, Image Polygonal Annotation with Python. Available online: https://github.com/wkentaro/labelme, 2016.

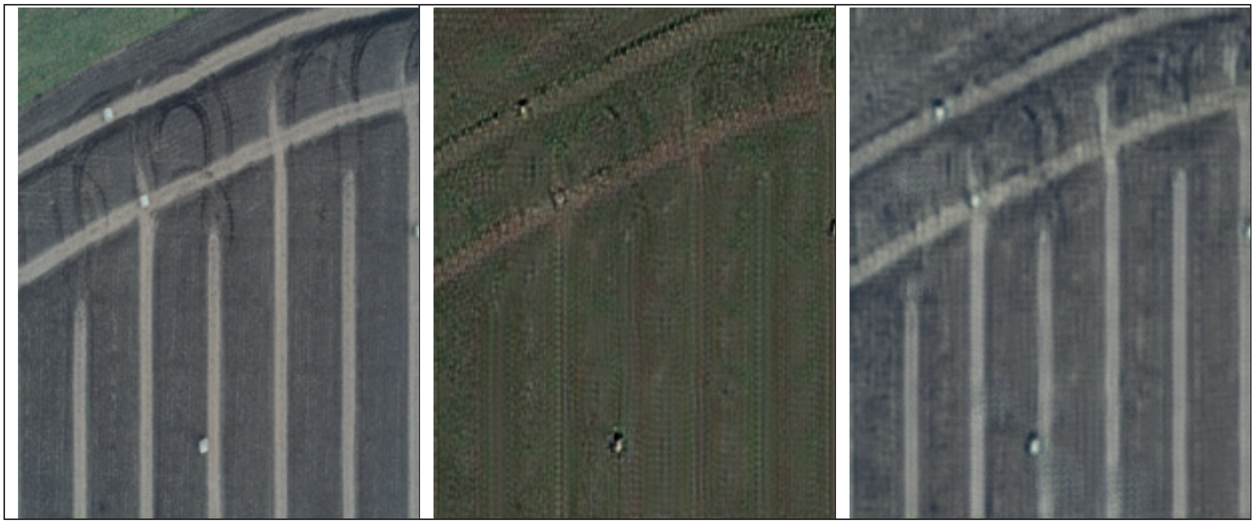
Appendix 1

Detailed CycleGAN result thumbnail images:

Real image	Corresponding generated fake image by transfer learning	Reconstructed real image using fake image
		
		

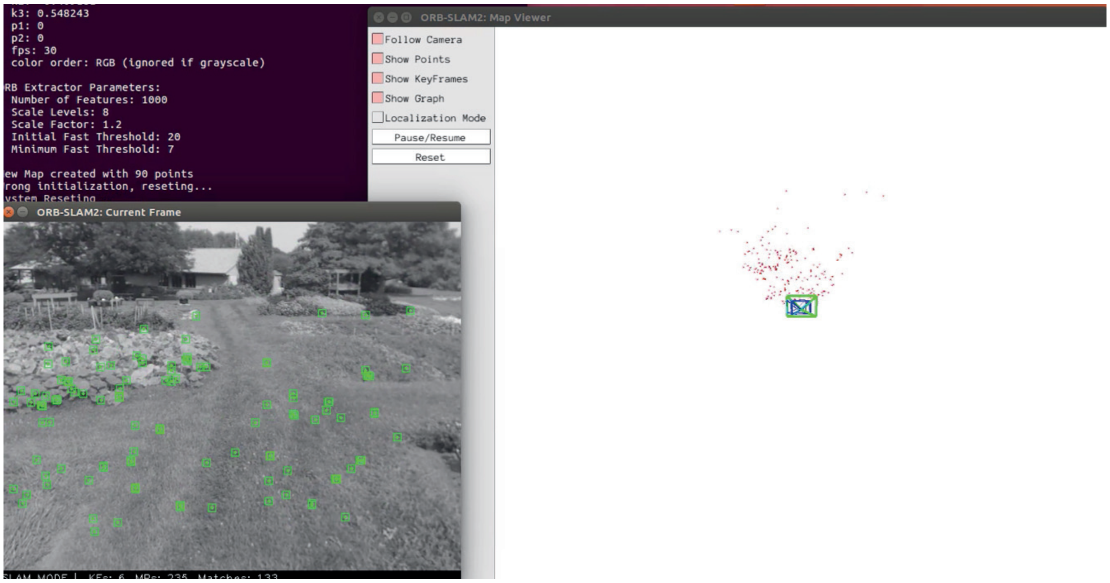






Appendix 2

The sample frames from the video when building the SLAM map

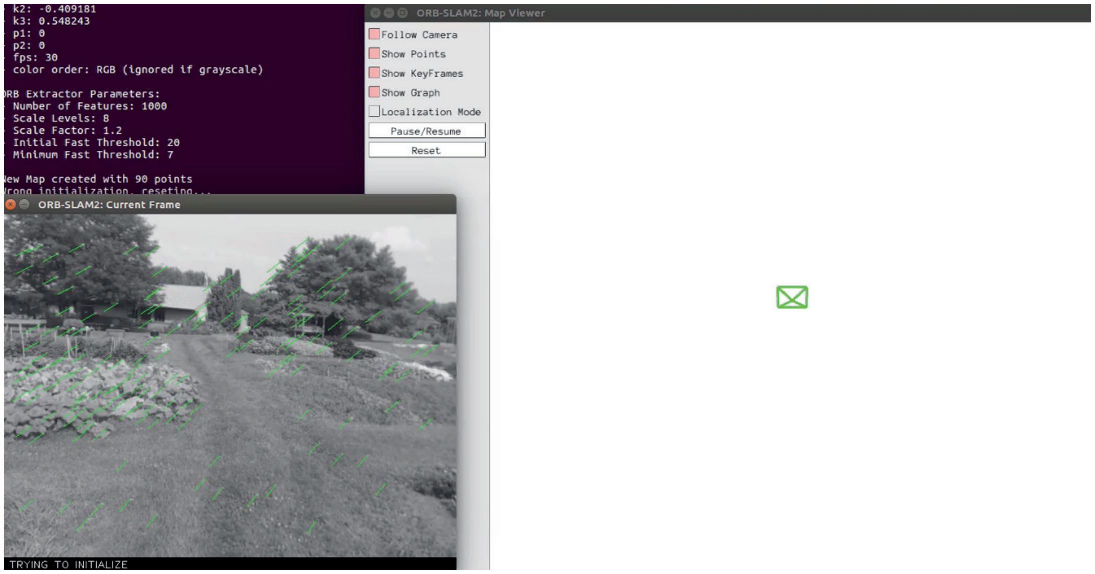


```
 k3: 0.548243
 p1: 0
 p2: 0
 fps: 30
 color order: RGB (ignored if grayscale)

RB Extractor Parameters:
 Number of Features: 1000
 Scale Levels: 8
 Scale Factor: 1.2
 Initial Fast Threshold: 20
 Minimum Fast Threshold: 7

ew Map created with 90 points
rong initialization, reseting...
ystem Reseting
```

```
SLAM MODE |  KFs: 6  MPs: 235  Matches: 133
```

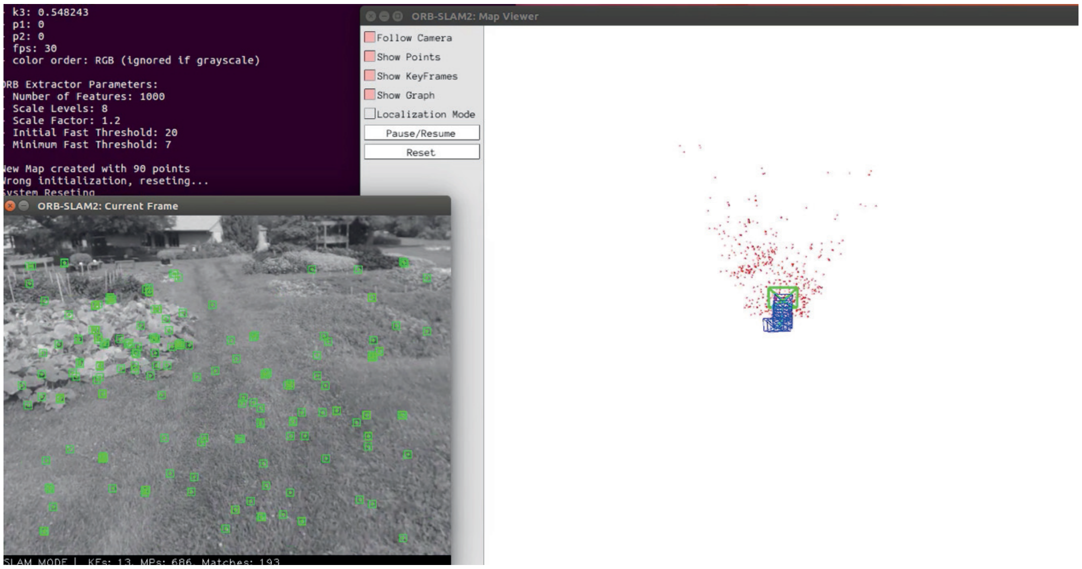


```
 k2: -0.409181
 k3: 0.548243
 p1: 0
 p2: 0
 fps: 30
 color order: RGB (ignored if grayscale)

ORB Extractor Parameters:
 Number of Features: 1000
 Scale Levels: 8
 Scale Factor: 1.2
 Initial Fast Threshold: 20
 Minimum Fast Threshold: 7

New Map created with 90 points
Wrong initialization, reseting...
```

```
TRYING TO INITIALIZE
```



```
- k3: 0.548243
- p1: 0
- p2: 0
- fps: 30
- color order: RGB (ignored if grayscale)

ORB Extractor Parameters:
- Number of Features: 1000
- Scale Levels: 8
- Scale Factor: 1.2
- Initial Fast Threshold: 20
- Minimum Fast Threshold: 7

New Map created with 90 points
Wrong initialization, reseting...
System Reseting
```

ORB-SLAM2: Map Viewer

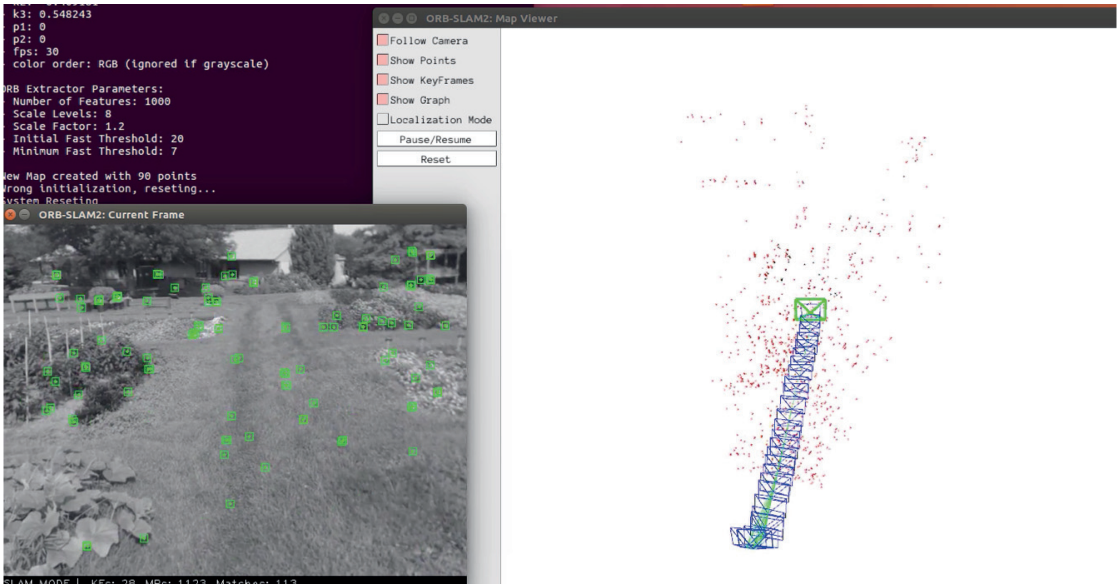
☑ Follow Camera
☑ Show Points
☑ Show KeyFrames
☑ Show Graph
□ Localization Mode

Pause/Resume

Reset

ORB-SLAM2: Current Frame

```
SLAM MODE |  KFs: 13, MPs: 686, Matches: 193
```



```
- k3: 0.548243
- p1: 0
- p2: 0
- fps: 30
- color order: RGB (ignored if grayscale)

ORB Extractor Parameters:
- Number of Features: 1000
- Scale Levels: 8
- Scale Factor: 1.2
- Initial Fast Threshold: 20
- Minimum Fast Threshold: 7

New Map created with 90 points
Wrong initialization, reseting...
System Reseting
```

ORB-SLAM2: Map Viewer

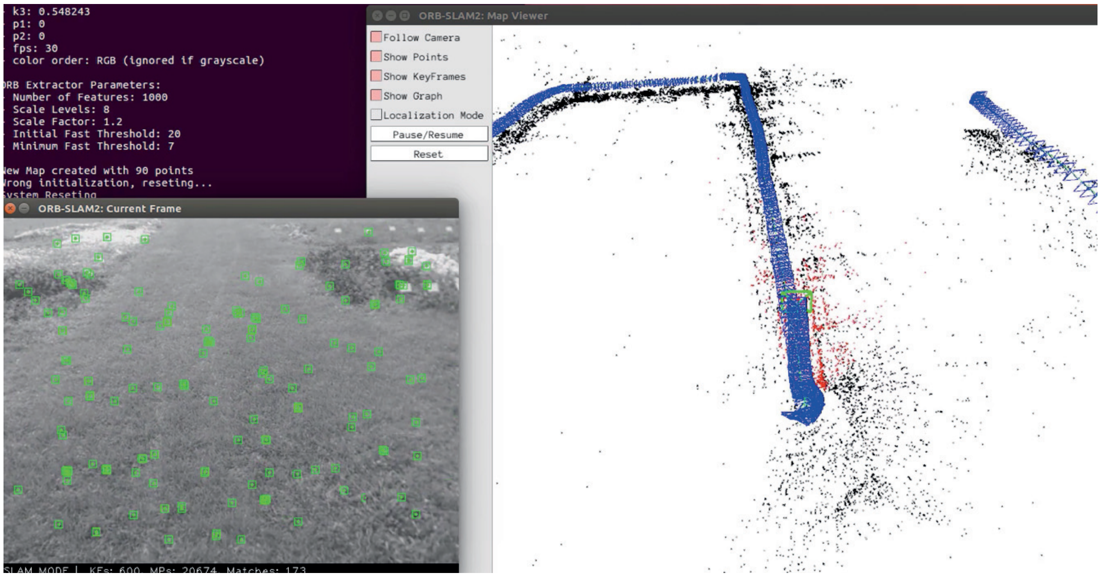
☑ Follow Camera
☑ Show Points
☑ Show KeyFrames
☑ Show Graph
□ Localization Mode

Pause/Resume

Reset

ORB-SLAM2: Current Frame

```
SLAM MODE |  KFs: 28, MPs: 1123, Matches: 113
```

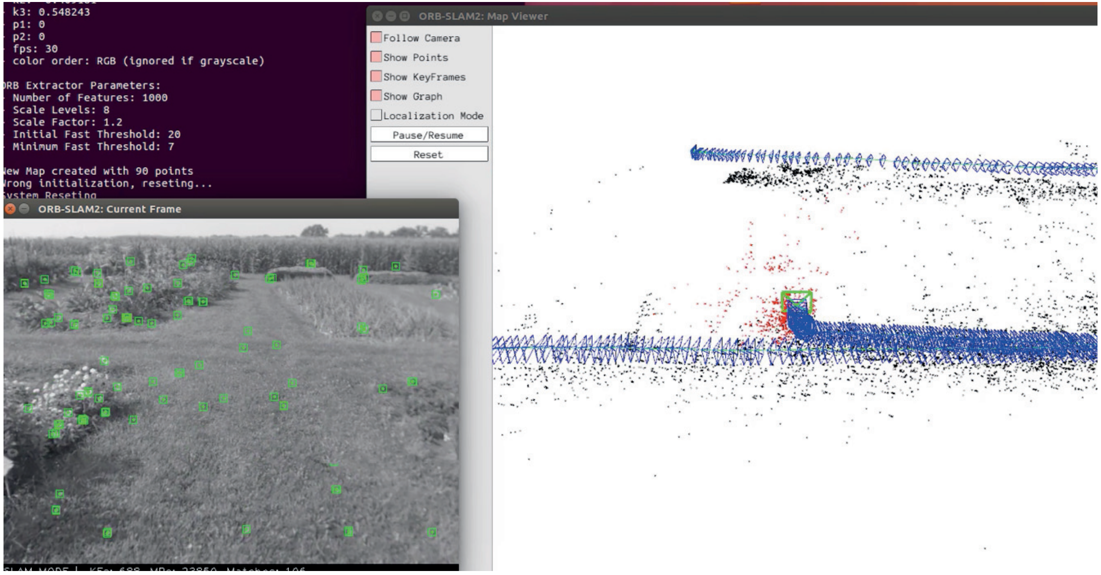


```
- k3: 0.548243
- p1: 0
- p2: 0
- fps: 30
- color order: RGB (ignored if grayscale)

ORB Extractor Parameters:
- Number of Features: 1000
- Scale Levels: 8
- Scale Factor: 1.2
- Initial Fast Threshold: 20
- Minimum Fast Threshold: 7

New Map created with 90 points
Wrong initialization, reseting...
System Reseting
```

```
SLAM MODE |  KFs: 600, MPs: 20674, Matches: 173
```



```
- k3: 0.548243
- p1: 0
- p2: 0
- fps: 30
- color order: RGB (ignored if grayscale)

ORB Extractor Parameters:
- Number of Features: 1000
- Scale Levels: 8
- Scale Factor: 1.2
- Initial Fast Threshold: 20
- Minimum Fast Threshold: 7

New Map created with 90 points
Wrong initialization, reseting...
System Reseting
```

```
SLAM MODE |  KFs: 688, MPs: 23850, Matches: 106
```