A HUB-CI MODEL FOR NETWORKED TELEROBOTICS IN COLLABORATIVE MONITORING OF AGRICULTURAL GREENHOUSES

by

Ashwin Sasidharan Nair

A Thesis

Submitted to the Faculty of Purdue University

In Partial Fulfillment of the Requirements for the degree of

Master of Science in Industrial Engineering

School of Industrial Engineering

West Lafayette, Indiana

May 2019

THE PURDUE UNIVERSITY GRADUATE SCHOOL
STATEMENT OF COMMITTEE APPROVAL

Dr. Shimon Y. Nof, Chair
Department of Industrial Engineering
Dr. Seokcheon Lee
Department of Industrial Engineering
Dr. Vaneet Aggrawal
Department of Industrial Engineering

Approved by:

Dr. Steven J. Landry
Head of the Graduate Program

To my family

ACKNOWLEDGMENTS

I would like to thank my thesis advisor Dr. Shimon Y. Nof for his guidance throughout the duration of my research. I would also like to thank Dr. Avital Bechar of Agricultural Research Organization, Rishon LeZion, Israel for his invaluable guidance in this work, and for providing spectral image data that was used in this research. I would also like to acknowledge the US-Israel Binational Agricultural Research and Development Fund (BARD) for providing the grant #4886-16R, *Development of a Robotic Inspection System for Early Identification and Locating of Biotic and Abiotic Stresses in Greenhouse Crops*, that funded this work. I would also like to acknowledge my thesis committee members Dr. Seokcheon Lee and Dr. Vaneet Aggarwal for their invaluable contributions to this effort.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

NOMENCLATURE

ANOVA - Analysis of Variance 67
ARS - Agricultural Robotic System 13
CCT - Collaborative Control Theory 24
CI - Collaborative Intelligence 24
CPS - Cyber Physical System 19
DSS - Decision Support System 14
GIS - Geographic Information System 17
HRI - Human Robot Interaction 15
HUB-CI - Hub with Collaborative Intelligence 22
PRISM - Production, Robotics, and Integration Software for Manufacturing & Management 18
ROS - Robot Operating System 18
SDIs - Spectral disease indices 29
SWIR - Short Wavelength Infrared 29
TTC - Time to Complete 51
VNIR - Visible and Near Infrared 29

ABSTRACT

Author: Nair, Ashwin, S. MS Industrial Engineering
Institution: Purdue University
Degree Received: May 2019
Title: A HUB-CI Model for Networked Telerobotics in Collaborative Monitoring of Agricultural Greenhouses
Committee Chair: Shimon Y. Nof

Networked telerobots are operated by humans through remote interactions and have found applications in unstructured environments, such as outer space, underwater, telesurgery, manufacturing etc. In precision agricultural robotics, target monitoring, recognition and detection is a complex task, requiring expertise, hence more efficiently performed by collaborative human-robot systems. A HUB is an online portal, a platform to create and share scientific and advanced computing tools. HUB-CI is a similar tool developed by PRISM center at Purdue University to enable cyber-augmented collaborative interactions over cyber-supported complex systems. Unlike previous HUBs, HUB-CI enables both physical and virtual collaboration between several groups of human users along with relevant cyber-physical agents. This research, sponsored in part by the Binational Agricultural Research and Development Fund (BARD), implements the HUB-CI model to improve the Collaborative Intelligence (CI) of an agricultural telerobotic system for early detection of anomalies in pepper plants grown in greenhouses. Specific CI tools developed for this purpose include: (1) Spectral image segmentation for detecting and mapping to anomalies in growing pepper plants; (2) Workflow/task administration protocols for managing/coordinating interactions between software, hardware, and human agents, engaged in the monitoring and detection, which would reliably lead to precise, responsive mitigation. These CI tools aim to minimize interactions' conflicts and errors that may impede detection effectiveness, thus reducing crops quality. Simulated experiments performed show that planned and optimized collaborative interactions with HUB-CI (as opposed to ad-hoc interactions) yield significantly fewer errors and better detection by improving the system efficiency by between 210% to 255%. The anomaly detection method was tested on the spectral image data available in terms of number of anomalous pixels for healthy plants, and plants with stresses providing statistically significant results between the different classifications of plant health using ANOVA

tests (P-value = 0). Hence, it improves system productivity by leveraging collaboration and learning based tools for precise monitoring for healthy growth of pepper plants in greenhouses.

1. INTRODUCTION

1.1 Background - Precision Agriculture, Automation in Greenhouse Agriculture, E-work and E-Systems, Collaboration Engineering

Precision agriculture is agricultural management system where practices for crop production and their corresponding inputs such as seed, fertilizers, pesticides etc. are variably applied within an agricultural area. Optimum production needs determine the input rates for resources at each specific field location (Sudduth, 1998). Precision agriculture techniques can improve the economic and environmental sustainability of crop production. Another term used to refer to precision agricultural techniques is precision farming, defined as the farm management strategy utilizing precise information and information gathering technology i.e. different types of sensors to increase profit and reduce environmental impact (An, Wu et al., 2017).

An agricultural environment is a complex and unstructured environment. Production in an agricultural environment is intensive and requires development of robust systems with short development time at low cost. The unstructured nature of the external environment increases chances of failure (Han, Edan, Kondo, 2009). The introduction of automation into agriculture has resulted in lowered production costs, reduced tedious manual labor, raised quality of fresh produce, and improved control of the production environment for crops. Industrial applications usually deal with simple, repetitive, well-defined, and a known a priori tasks. This differs from automation in agriculture which requires advanced technologies to deal with the complex and highly variable environment and produce (Edan, Han, Kondo, 2009). Automation increases the productivity of agricultural machinery via increased efficiency, reliability, and precision, and minimal human intervention all of which is achieved by adding sensors and controls (Edan, Han, Kondo, 2009).

Greenhouses have been developed through the 20th century to contain energy from solar radiation, to protect products from various hazardous natural climates and insects, and to produce suitable environments for growing plants (Edan, Han, Kondo, 2009). The greenhouse environment is a relatively easy environment for introduction of automated machinery due to its structured nature. Automation systems for greenhouses deal with tasks like climate control, seedling production, spraying, and harvesting (Edan, Han, Kondo, 2009). The research for this thesis was undertaken as part of an effort to create an agricultural robotic system (ARS) for disease detection of pepper plants in greenhouses.

Here are some other terms and concepts that are part of this research that need to be defined: *HUB* as defined in this research is an online portal built on the HUBzero technology to support collaborative development and sharing of scientific models and tools that are running in a cloud-based infrastructure of computing resources (McLennan, Kennell, 2010). *Cyber Physical Systems* (CPS): CPSs are commonly defined as the systems which offer collaborative integrations of computation, networking, and physical processes (Khaitan et al., 2015). As per the US National Science Foundation, "In cyber-physical systems, physical and software components are deeply intertwined, each operating on different spatial and temporal scales, exhibiting multiple and distinct behavioral modalities, and interacting with each other in a myriad of ways that change with context." *e-Work* is a collection of collaborative, computer-supported, and communication-enabled e-Activities, e-Operations, e-Functions, and e-Support systems that enables other e-Systems and e-Activities (Nof, 2003). *Decision Support System (DSS)* is generally an auxiliary interactive computer-based system that leverages computer communications, data, knowledge, and relevant models to identify and solve problems, and complete various decision process tasks (Nof et al., 2015). *Agents* are defined as independent and

autonomous programs, which operate and execute specific tasks under certain protocols, and are responsible for handling expected and unexpected events (Nof et al., 2015). *Collaboration engineering* as defined by Nof et. al 2015, refers to the "development of tools and techniques to go beyond better communication technologies and provide significantly improved collaboration support by cyber through algorithms, protocols, and collaboration support software agents."

Agricultural robotic applications do require advanced technologies to be productive and to handle complex environments with high variability (Nof, 2009) and not all agricultural applications can be fully automated in the near term. However, partial autonomy with collaboration can add value in terms of efficiency and productivity and its capabilities before full autonomy is achieved. In a target recognition task collaboration between a human operator and a robot increases probability of detection by 4 percent when compared to a HO alone and by 14 percent when compared to a fully autonomous system (Bechar, Edan, 2003).

1.2 Research Objectives

The problem addressed in this research is how to create a collaborative human robot greenhouse crop management system. This research aims to develop a human-robot collaborative system based on the concepts of HUB, Decision Support System (DSS), and Collaboration Engineering to improve the process of disease detection of pepper plants in greenhouses with spectral imaging sensors, and automated robot navigation processes. The objectives of this research include:

1) To create a HUB based Human Robot Interaction (HRI) mechanism that facilitates physical and virtual collaboration between a) Agricultural Experts, b) Human operators, c) Multiple software agents related to navigation, control and disease detection, all of which are not at the same geographical location (remote agents/operators).

2) To develop learning-based protocols to enhance Human Robot Interaction and Decision Support capabilities of the Integrated Planner
3) To enable early detection of anomalous pepper plants likely to have biotic or abiotic stress.

1.3 Research Questions

The following research questions are addressed in this work:

1. How to design a collaborative e-system comprised of remote agents (human, machine and software) for the detection and identification of anomalies in pepper plants in a greenhouse setting such that the system performance is optimized and robust for error?
2. What task administration protocols are necessary for collaborative control of this system?
3. What DSS tools are necessary for a) optimal collaboration and minimal error in an agricultural setting, and b) enable early detection of stresses in plants?

1.4 Significance of Research

This research attempts to address the following questions/issues:

1) How to build an optimal agricultural greenhouse system that combines the cognitive and perceptive capabilities of a human agent, with the precision, speed and consistency of a robot, in a way that optimizes the output of the greenhouse?
2) Use of learning-based Decision Support Tools to improve collaboration and enhance the Collaborative Intelligence of the system
3) Creating a strategy for Human-Robot collaboration for greenhouse monitoring and simulating it for effectiveness.

These are key issues that will help create the future Agricultural Systems that leverage both human intelligence and machine intelligence in a symbiotic and productive manner.

2. LITERATURE REVIEW

2.1 Collaborative Robotics in Agriculture

Unstructured environments such as agriculture are characterized by rapid changes in time and space unlike in industrial environments where working conditions are usually fixed and predictable (Bechar et al., 2003). Humans have superior and acute perception capabilities which enables them to work in a broad scope of relatively vague and unstructured conditions (Chang, Song, Hsu, 1998). Moreover, humans have superior recognition capabilities and can adapt easily to changing environmental and object conditions (Rodriguez, Weisbin, 2003). However, a human operator is not consistent, prone to fatigue, and is subject to distraction. Bechar et al., 2009 developed an objective function designed to allow computation of the expected value of system performance, given the parameters of the overall system, the task, and the environment. It evaluates statistically the value of a specific system in a specific task and includes the direct gains, rewards, costs, and penalties of such a system (Bechar et al., 2009). Cheein et al., 2015 performed a study that surveyed Human Robot Interaction practices in harvesting and included guidelines for designing a human-robot interaction strategy for harvesting tasks, which could also be used for other agricultural tasks. As per that research, the four design cores of a service unit are: mapping, navigation, sensing and action. This research was addressing the problem of declining human labor force in agriculture in the countries of Chile and Argentina. That study discussed the advantages and challenges associated with flexible automated farming.

2.2 Simulating an Agricultural Robotic System

As more robotic systems are being developed and implemented in the field of agriculture, it would be cost effective to simulate such systems in the development phase. Recently there have been a few research projects on simulating a robotic system for Human-Robot collaboration. A computational simulation environment named "Simulation Environment for Precision Agriculture Tasks using Robot Fleets" (SEARFS) was developed (Emmi et al., 2013) to study and evaluate the execution of agricultural tasks that can be performed by an autonomous fleet of robots. The environment is based on a mobile robot simulation tool that enables the analysis of performance, cooperation, and interaction of a set of autonomous robots while simulating the execution of specific actions on a three-dimensional (3D) crop field. The environment is capable of simulating new technological advances such as a GPS, a GIS, automatic control, in-field and remote sensing, and mobile computing, which will permit the evaluation of new algorithms derived from Precision Agriculture techniques. This environment was designed as an open source computer application. The SEARFS environment consists of four levels of configurations, where the lower levels depend on the configuration of the higher levels.

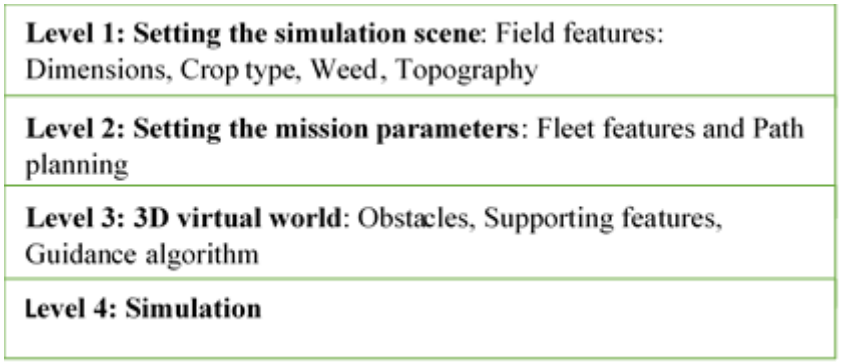


Figure 2-1 SEARFS environment configuration levels (Courtesy of Emmi et al., 2013)

A general method for development of customized robot simulation and control system software with robot operating system (ROS) was also developed (Wang, et al., 2016). The simulation designed in this research involves a) A 3D visualization model, created in URDF (unified robot description format) and viewed in Rviz to achieve motion planning with MoveIt! software package; b) A machine vision provided by camera driver package in ROS to enable the use of tools for image processing, and 3D point cloud analysis to reconstruct the environment to achieve accurate target location; and c) Communication protocols provided by ROS for serial, Modbus support of the communication system development. A tomato harvesting scenario was simulated using this methodology to demonstrate its features and effectiveness.

2.3 Collaborative Control Theory

Collaborative Control Theory has been developed by researchers at the PRISM center at Purdue University and elsewhere (Nof et al., 2015) to optimize distributed, decentralized, and multi-agent-based e-Work and s-Service. Collaboration is known to be an essential means for effective design and control of e-Work and e-Service. Collaboration enables all involved entities, human and artificial, in decentralized e-Systems to share their resources, information, and responsibilities, such that mutual benefits are obtained.

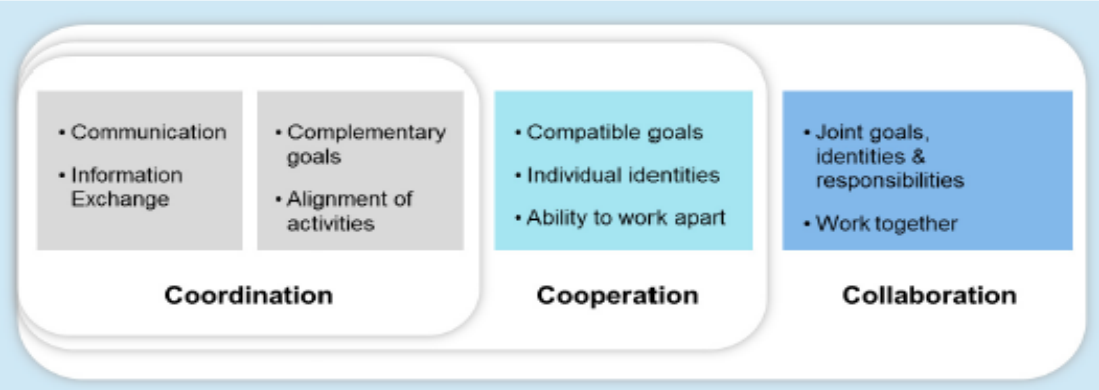


Figure 2-2 Coordination vs Cooperation vs Collaboration in terms of interaction level (Source: Nof et al., 2015)

As future precision agricultural systems are likely to be comprised of multiple distributed and autonomous agents, the efficiency and effectiveness of the CPS would depend upon how well its constituent agents can collaborate.

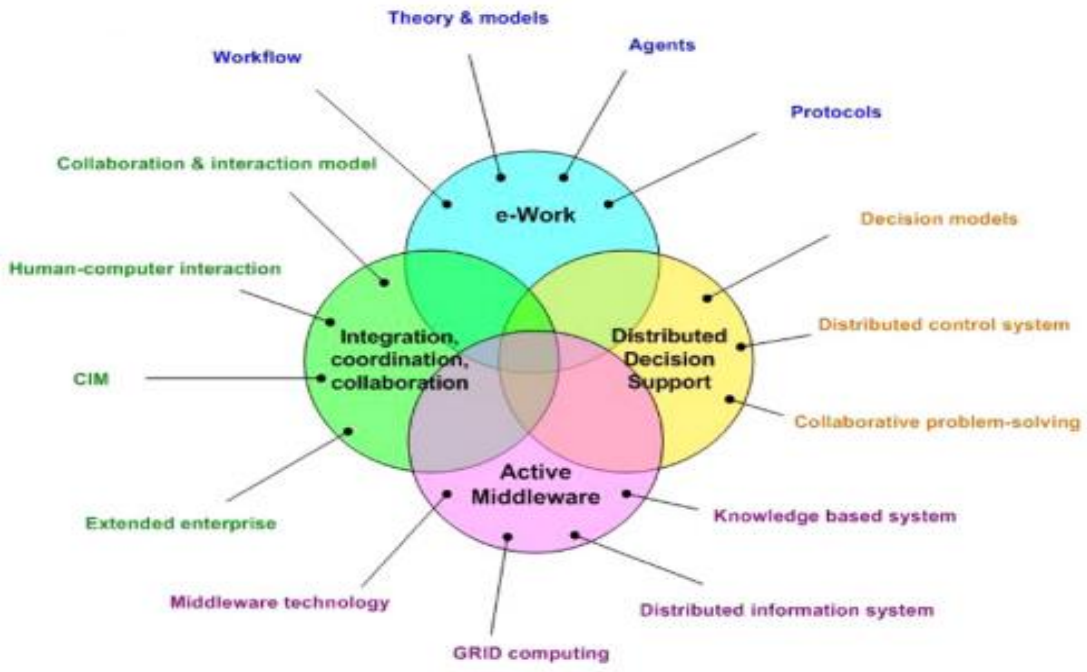


Figure 2-3 The different components of cyber enhanced processes i.e. e-Work

Below is a summary of the basics of Collaborative Control Theory (Nof et al., 2015):

1) Collaboration Requirement Planning (CRP): The first step necessary is to make use of planned collaboration, as opposed to ad hoc or unplanned collaboration. Effective collaboration requires advanced planning and feedback loops, as targeted by the CRP principle. CRP is composed of two phases namely Plan Generation, and Plan Execution and Revision.
2) E-Work Parallelism (EWP): The EWP principle is based on the fact that in any e-System, as a distributed network of agents, some activities can and should be performed in parallel. The principle of EWP has deeper implications due to the distributed nature of the e-Systems and the fact that interactions take place through human and software workspaces, including 1) human-human interactions, 2) human-machine and human-computer interactions, and/or 3) machine-machine and computer-computer interactions.
3) Keep It Simple, System (KISS): Any e-System can be as complicated as need be, as long as it is as simple as possible for human participants' interactions. This principle builds on and embraces traditional human-computer and human-automation usability design principles and goes beyond them.
4) Error Prevention and Conflict Resolution (EPCR): This principle deals with the detection of errors and conflicts among collaborating agents, and the costs associated with resolving the detected errors and conflicts. Naturally, any system that cannot overcome effectively its errors and conflicts will get out of control and eventually collapse. In general, the rates of errors and conflicts among agents are proportional to the rate of interactions and the number of active collaborating agents. Hence, effective collaboration requires timely detection and resolution of errors and conflicts as economically as

possible. The EPCR principle is composed of eight consecutive functions that begin with detection and end with resolution and exception handling. These include 1) Detection i.e. searching for existing Errors/Conflicts, 2) Identification i.e. classification of the observation as an error or conflict, 3) Isolation i.e. determining the exact point of an error or conflict in the system, 4) Diagnostics of Error/Conflict, 5) Prognostics, 6) Error recovery i.e. removing or mitigating the effects of an error, 7) Conflict Resolution, 8) Exception handling i.e. managing exceptions in the process.

5) Collaborative Fault Tolerance (CFT): The purpose is to achieve higher efficiency and reliability from a network of weak agents (e.g., micro-sensors) rather than a single stronger agent.
6) Association and Dissociation (AD): This principle addresses dynamic variations in the formation (topology), size, and operations of collaborative networks of agents. The AD principle analyzes the conditions and timing for individual agents or networks of agents to associate with or disassociate from a collaborative network.
7) Emergent Lines of Collaboration and Command (ELOCC): The purpose of this principle to overcome drastic changes facing networks, e.g. under emergency, and the volatility of formal and informal communications between the individual and clustered agents.
8) Best Matching (BM): Matching between the collaborative agents is a fundamental concern in the design and execution of collaborative e-Work networks. The objective of the BM principle is to find the best match between two or more sets of agents, such that a set of objectives is satisfied.
9) Collaborative Visualization and Comprehension (CVC): Visual analytics focuses on the integration of interactive visualization with analytic tools and techniques to deal with the

rising complexities of e-Work systems. The aim is to integrate computer graphics, interaction, visualization, analytics, perception, and cognition domains to enhance and support the human machine interactions.

2.4 HUB-based Telerobotic Systems

With decreasing prices and fast paced advances in ubiquitous computing, telerobotics is gaining popularity as an attractive framework that allows true physical collaboration among distributed users (Song et al., 2008). Telerobotics can be seen as a form of e-work, which is a collaborative, computer-supported, and communication-enabled platform for operations in highly distributed organizations (Nof 2003).

What is HUB-CI and how can it improve an Agricultural Robotic System (ARS)?

HUB-CI was inspired from the research and implementation of HUBZero system developed by researchers at Purdue university. The HUBzero cyberinfrastructure, facilitates researchers to work together online to develop simulation tools (McLennan et al., 2010). Collaborators can access the resulting tools by ordinary Web browser and launch simulation runs on the national Grid infrastructure, without having to download any code. The Pegasus Workflow Management System can manage workflows comprising millions of tasks and recording data about their execution simultaneously (McLennan 2015). HUB is an online portal that provides users to create and share research materials and computational tools. HUB can deliver all resources and simulations via a regular web browser and utilize high performance Grid computing resources (McLennan et al., 2010). HUB along with cloud computing allows software and data to be easily shared among groups of users. Most HUBs allow collaborative jobs on virtual materials and simulations, but there was no tool for users to perform physical collaboration (McLennan et al., 2010). The innovation of HUB-CI is that it addresses managing

physical collaboration between groups of human users plus relevant cyber physical agents (Zhong et al., 2014). HUB-CI is a continuous project aimed at improving the collaboration methods over HUB-like systems. HUB-CI i.e. HUB-based Collaborative Intelligence is a set of collaborative intelligence algorithms and tools have been developed to enhance HUB and augment productivity with more efficient functions to support collaborations (Zhong et al., 2014).

So, for an Agricultural Robotic System (ARS), the aspect of physical along with virtual collaboration is where HUB-CI could add the most value in terms of augmenting efficiency and productivity.

The Networked Telerobot System for Agricultural Robotics over HUB involves multiple operators, single or multiple robots, and cyberinfrastructure to support collaboration.

Collaborative control is a process that is fault tolerant, and its benefit is that it can yield better results from a team of weak agents when compared to a system that depends on an agent that is faultless. (Nof et al., 2015).

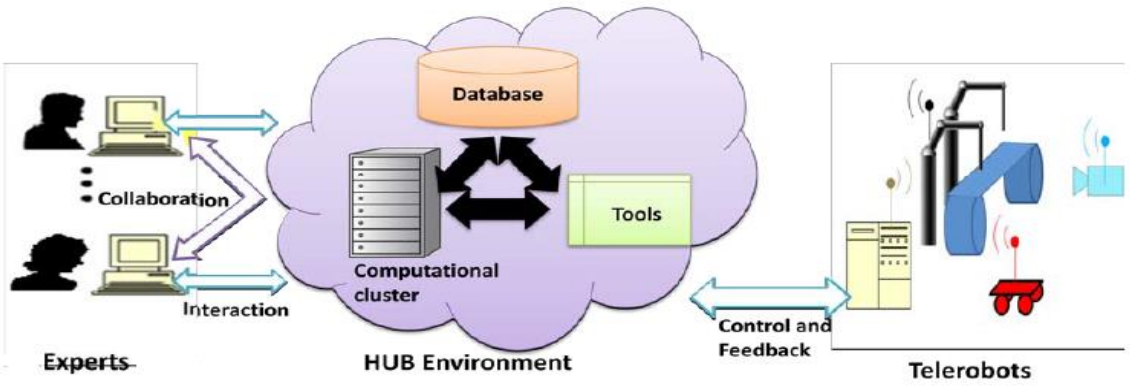


Figure 2-4 Architecture of HUB-based NTR Systems (Zhong, 2012)

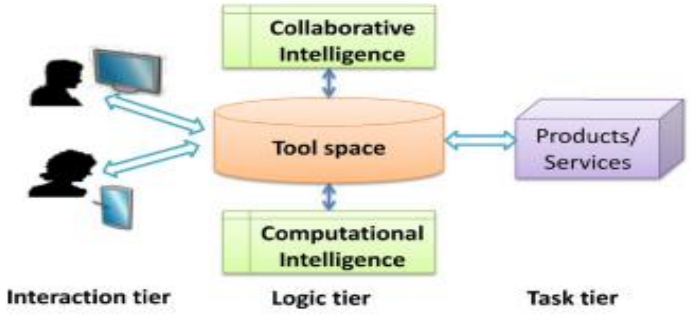


Figure 2-5 HUB-CI model for three-tier collaboration infrastructure (Zhong et. al 2014)

The HUB-CI used in this research follows the same high-level concept as that depicted in Fig. 2-5. The purpose of the HUB-CI is to serve as a) The Integrated Planner, b) Human Robot Interaction interface and Decision Support System.

2.5 Collaborative intelligence

Collaborative intelligence (CI) is a concept developed at PRISM center, Purdue University. With regard to Collaborative Control Theory (CCT) described earlier, the processes of collaborative E-Systems can be improved building and augmenting the Collaborative Intelligence (CI) of participants which can provide better support for achieving their individual and common goals (Zhong et al, 2015). Collaboration a necessary foundation for sustainability and evolution of any organization of natural or artificial agents, including cyber-physical systems (Nof et al., 2015). As seen in Fig. 2-2 regarding the differences between coordination, cooperation and collaboration, collaborative intelligence is a metric that incorporates and combines measures regarding all three. Zhong et al., 2015 defines collaborative intelligence as "CI is a measure of an agent's capability to perceive and comprehend new information, share

required resources, information, and responsibilities with other peers to resolve new local and global problems in a dynamic environment.

Figure 2-6 Interdependence of interaction intelligences (Courtesy: Devadasan et al., 2013)

Devadasan et al., 2013 collaborative intelligence (CI) in the knowledge-based service industry and to identify measures for finding the best collaborators during the formation and functioning stages of collaborative networks.

2.6 Agricultural Cyber Physical System framework for greenhouses

Guo, Dusadeerungsikul, Nof 2017 presented a CPS oriented framework and workflow for agricultural greenhouse stresses management, called MDR-CPS which has been designed to focus on monitoring, detecting and responding to various types of stress. The system combines sensors, robots, humans and agricultural greenhouses as an integrated CPS, for monitoring, detecting, and responding to abnormal situations and conditions aiming to provide an innovative solution combined wireless sensor networks, agricultural robots, and humans based on collaborative control theory in order to detect and respond to detected stresses as early as possible. Figure 2-7 describes the Agricultural CPS framework (Guo, Dusadeerunsikul and Nof 2017):

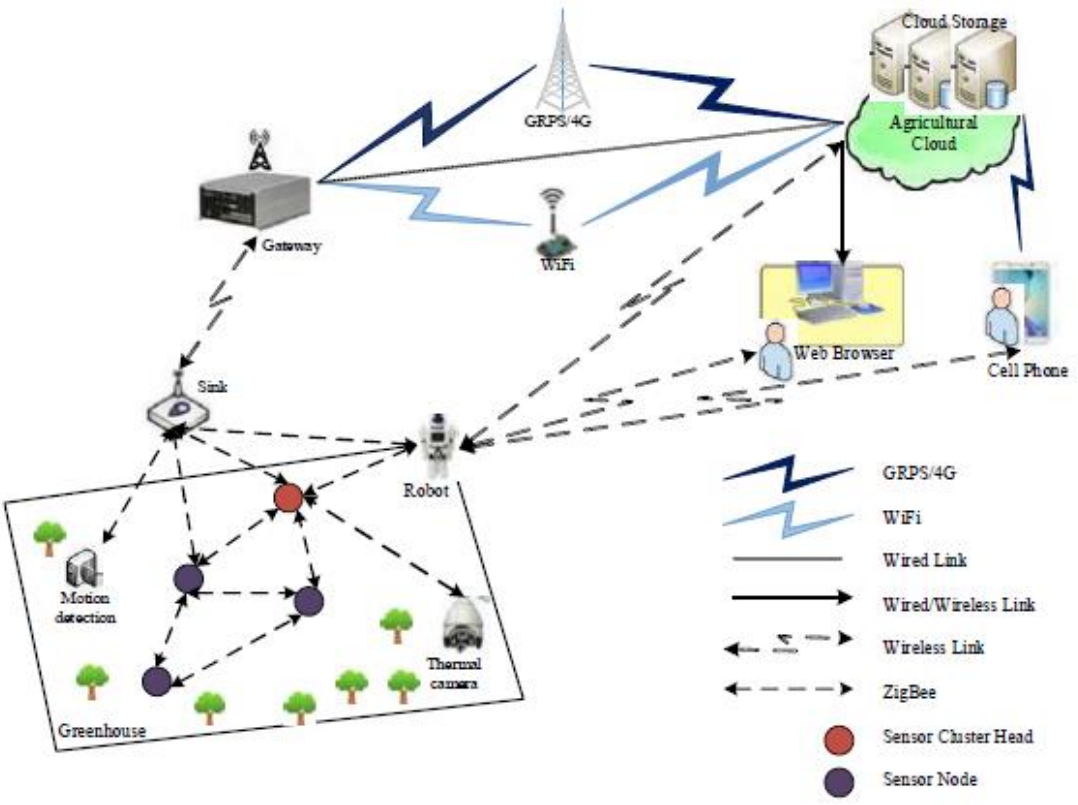


Figure 2-7 Agricultural CPS Framework (Courtesy of Guo, Dusadeerunsikul, Nof, 2017)

Sensors' nodes are deployed in the greenhouses to provide information of environmental parameters that influence the development of the agricultural crops. An agricultural cloud model platform is used in the agricultural field based on a number of server clusters (Guo et al., 2017). It contains two components which are cloud storage and cloud computing/expert systems, and not only stores large amounts of sensing data, but also provides services such as crop diseases analysis, intruders' alarm, and stresses identified. Furthermore, the network layer provides routing and data aggregation services. The gateway connects the agricultural cloud by GPRS/4G, Internet, WiFi, or local area networks. Users or farmers can access agricultural data through web browser or smart phone. The agricultural robot is used to aid detection in special situations for special stresses. Though sensors can do much of the monitoring work, and also can obtain

pictures or photos, they are limited by power, fixed location, and constraint transmission ability. The robot computer is to run the necessary software for interfacing to the robot platform and sensors, sensor information processing, mission planning and execution, navigation, implementation control, user interface, network communication, etc.

Below are the MDR-CPS workflow diagrams (Guo et al., 2017):

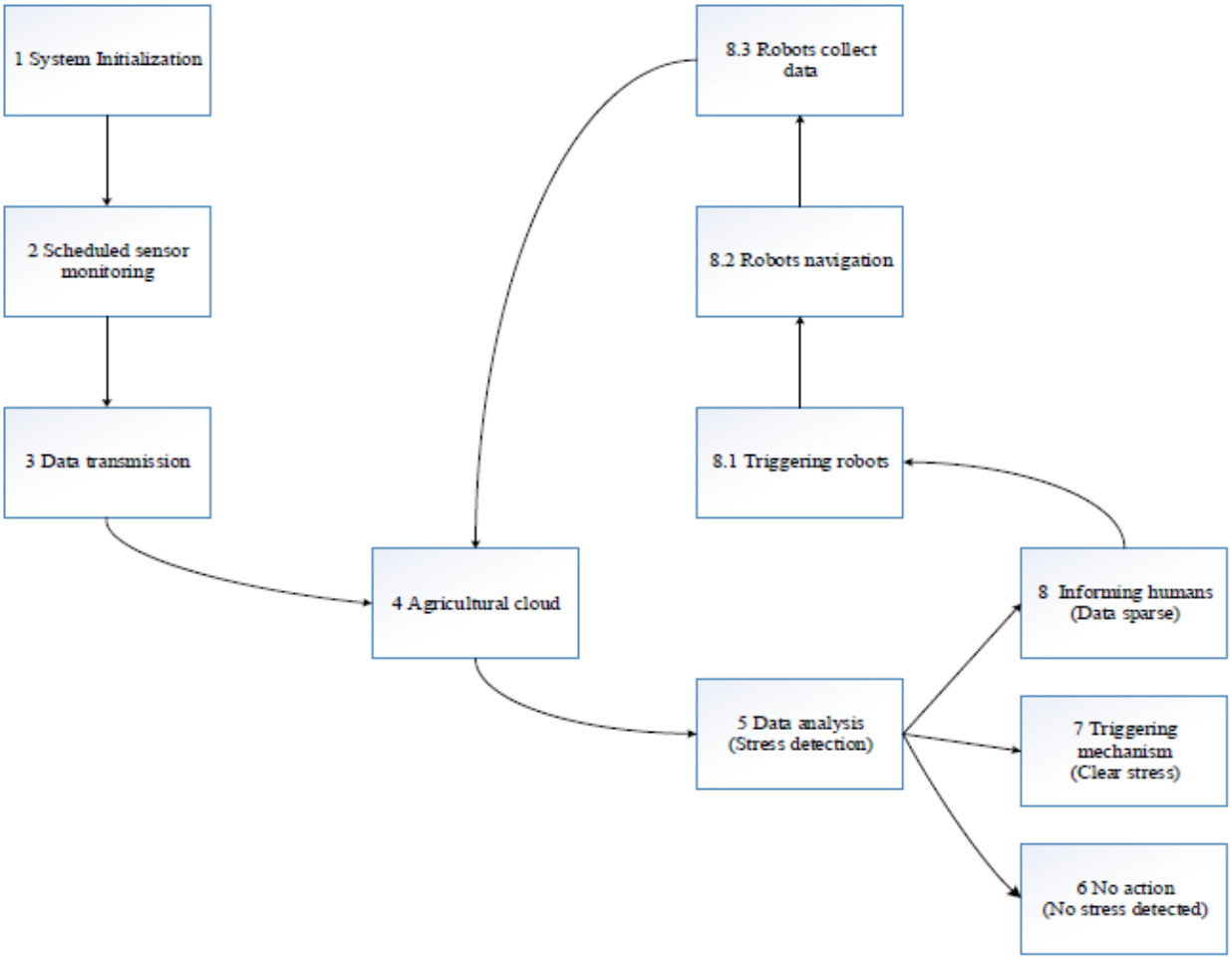


Figure 2-8 MDR-CPS workflow diagram (Courtesy of Guo, Dusadeerunsikul, and Nof, 2017)

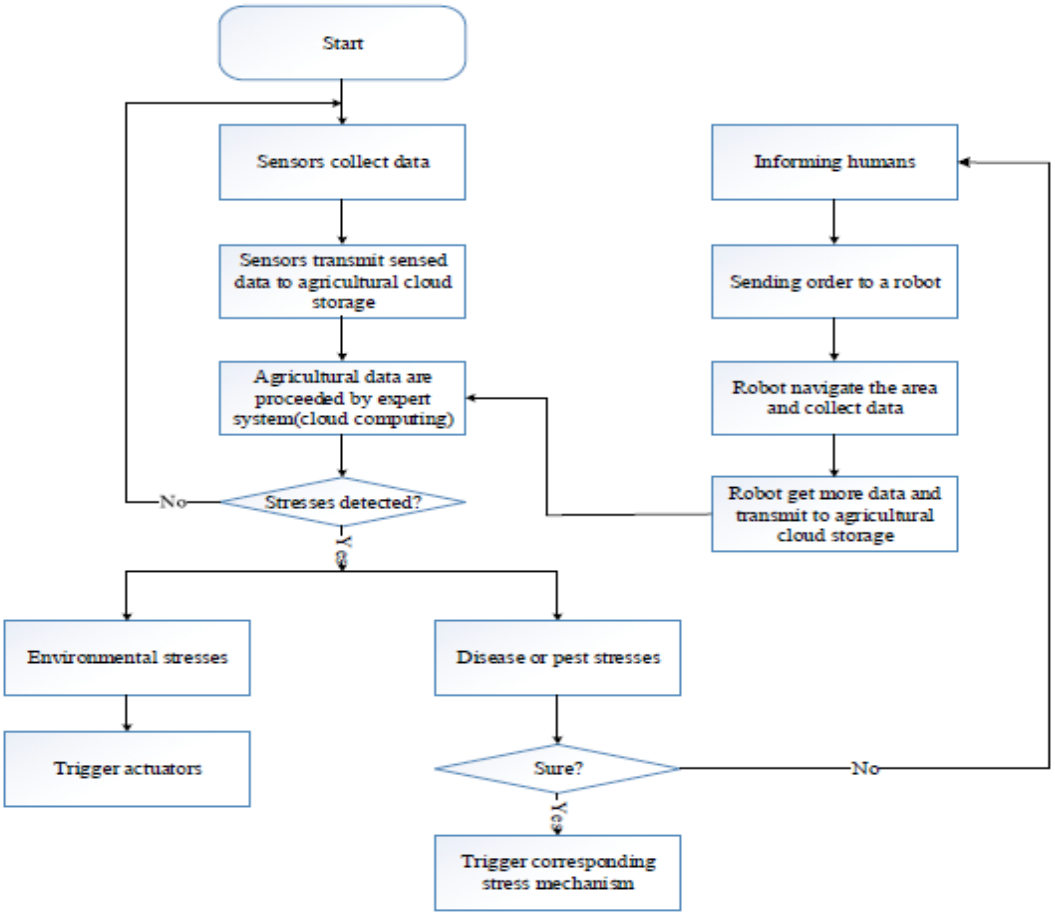


Figure 2-9 MDR-CPS workflow chart (Courtesy of Guo, Dusadeerunsikul, and Nof, 2017)

2.7 Spectral imaging on plants for stress detection

There have been several studies performed in the 21st century on the application of hyperspectral imaging for agricultural plants. Hyperspectral imaging has been applied successfully in plant disease classification and detection (Moghadam, Ward et al., 2017). A few general findings using hyperspectral images on plants include (Moghadam, Ward et al., 2017):

- A plant's interaction with different parts of the electromagnetic spectrum depends on leaf biochemical compounds and leaf anatomical structure. Healthy plants typically absorb light in the visible range (VIS 400-700 nm) due to leaf photosynthesis pigments.

- Amount of light scattered in the near-infrared range (NIR 700-1000 nm) is strongly sensitive to leaf cell structure.
- Factors that influence leaf reflectance in short-wave infrared (SWIR 1000-2500 nm) include leaf water and chemical contents.

Moghadam et al., 2017 used hyperspectral imaging (VNIR and SWIR) and machine learning techniques like feature extraction, cluster analysis and k-means for detection of the Tomato Spotted Wilt Virus (TSWV) in capsicum plants. Mahlein et al., 2013 developed specific spectral disease indices (SDIs) for the detection of diseases in crops specifically sugar beet plants with regard to three major leaf diseases of sugar beet plants. One key takeaway from that study relevant to this thesis is: Efficient use of spectral reflectance measurements for disease detection relies on identifying the most significant spectral wavelength which correlates strongly to a specific disease (Mahlein et al., 2013). A few regions of the spectrum are of interest and this depends on the application. Several studies developed advanced algorithms based on machine learning and image processing to determine plant part features and improve accuracy of monitoring, classification and feature extraction (AlSuwaidi et al., 2018; Nansen et al., 2013; Cheng et al., 2013). Wang, Vinson et al., (2018) developed a hyperspectral imaging technique for detection and classification of the plant disease TSVW using Generative Adversarial Nets, outlier removal, deep learning techniques which provided sensitivity and specificity of the classification at 92.59% and 100% respectively.

2.8 Summary of literature review

Table 2.1: Literature review and corresponding research questions

Research Question	Literature surveyed
How to model a collaborative e-system that accommodates remote agents (human, machine and software) for the detection and identification of anomalies in pepper plants in a greenhouse setting?	Guo et al., 2017; Emmi et al., 2013; Zhong et al., 2015; Nof et al., 2015; Wang, et al., 2016; Devadasan et al., 2013; Nof 2003; Bechar et al., 2003
What task administration protocols are necessary for collaborative control of this system?	Nof et al., 2015; McLennan et al., 2010; Zhong et al., 2014; Zhong, 2012; McLennan 2015; Bechar et al., 2009
What DSS tools are necessary for a) optimal collaboration with minimal error in an agricultural setting, and b) enable early detection of stresses in plants?	AlSuwaidi et al., 2018; Nansen et al., 2013; Cheng et al., 2013; Moghadam, Ward et al., 2017; Mahlein, Rumpf et al., 2013, Wang, Vinson et al., 2018

3. METHODOLOGY

3.1 HUB-CI in an Agricultural robotics system

With regards to previous research regarding the usefulness of Human-Robot collaboration as a productivity multiplier, this research aims to create a model that optimizes the collaborative capability of this multi-agent agricultural robotic system for the greenhouse. This is where the HUB-CI is relevant. HUB-CI serves as a Decision Support System and a Resource management tool. For the tasks in an agricultural robotic system that requires a human-in-the-loop, an experimental platform and a simulation of the concept of HUB-CI (hub for Collaborative Intelligence) system has been developed for this research to incorporate agents from other teams for collaboration that may need more than simple information sharing. The objective: Enabling effective integration and collaboration tasks, exchanging and leveraging collaborative intelligence from the ARS networked components whose physical location may be local or remote.

3.2 Agents based models of the ARS HUB-CI

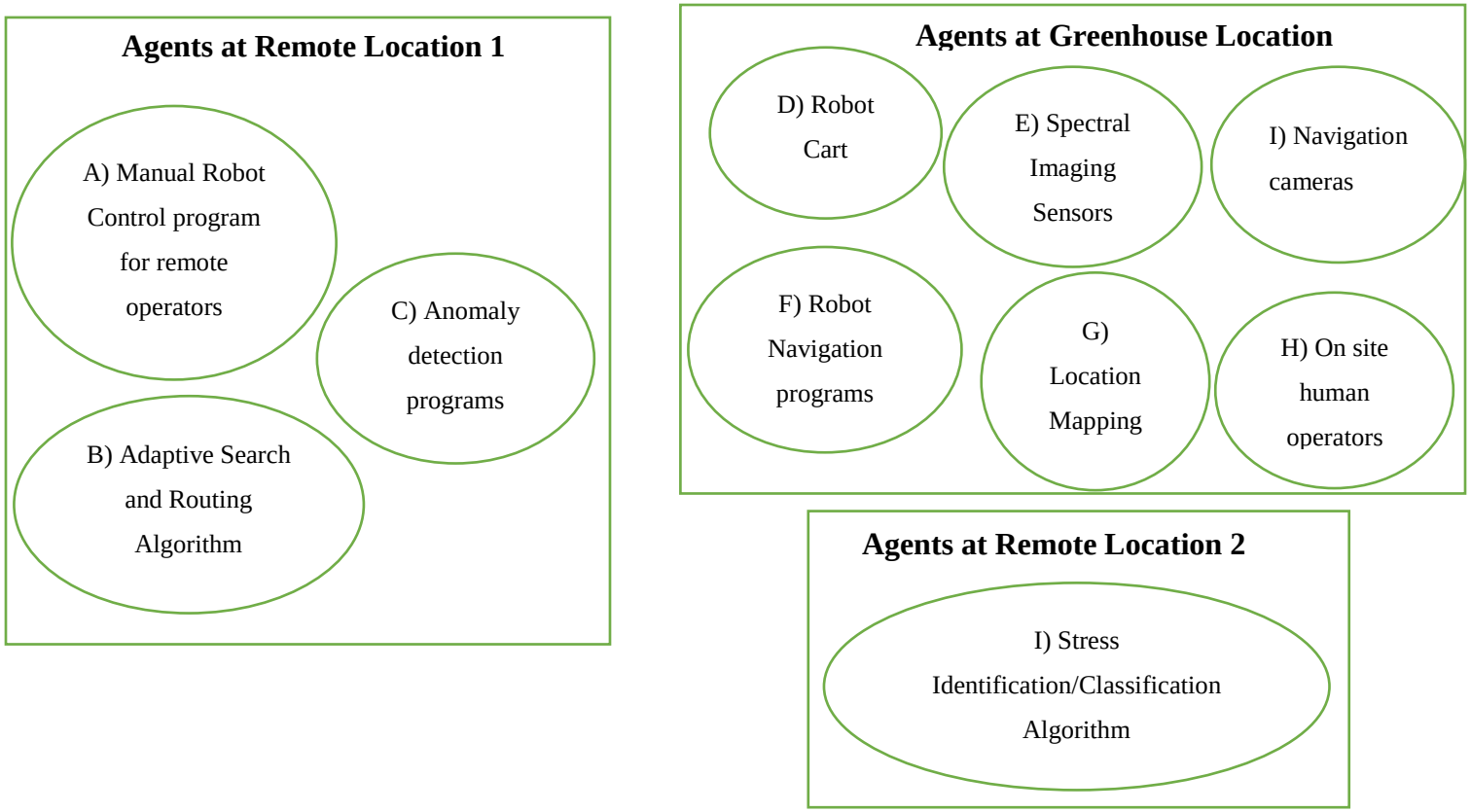


Figure 3-1 Agent based model for all hardware and software agents for the ARS HUB-CI system based on location

Table 3.1: Inputs and Outputs for agents in an Agricultural Robotic System

Agent	Input	Data type	Output	Data type
A	Direction commands (via keyboard)	String	Direction commands	String
B	1) Disease name 2) Direction of disease propagation 3) Distance matrix between nodes in greenhouse	1) String 2) String 3) Float	Near optimal routing sequence for robot cart.	Array
E	-	-	Spectral Images	m*n*k array
F	1) Map 2) Direction commands	Array, String	1) Robot odometry 2) Robot Pose (3D position and 3D orientation)	Array
G	Navigation sensor information		Map of greenhouse	
I	Spectral Images	m*n*k array	Stress diagnosis	String

Table 3.1 provides an example of inputs and outputs that are expected to be part of the agricultural robotic system. An important aspect of the HUB-CI is facilitating communication via data, information, knowledge and logic transfer across multiple agents in the system. It determines which agents must collaborate on which task or set of tasks. From the standpoint of developing the software, the above table helps define expected data types for each of the inputs and outputs.

How these agents communicate and collaborate with each other i.e. under what set of protocols should they communicate is illustrated in Figure 3-2. Figure 3-2 is a representation of the HUB-CI system based on its functions and the interactions between agent-based systems.

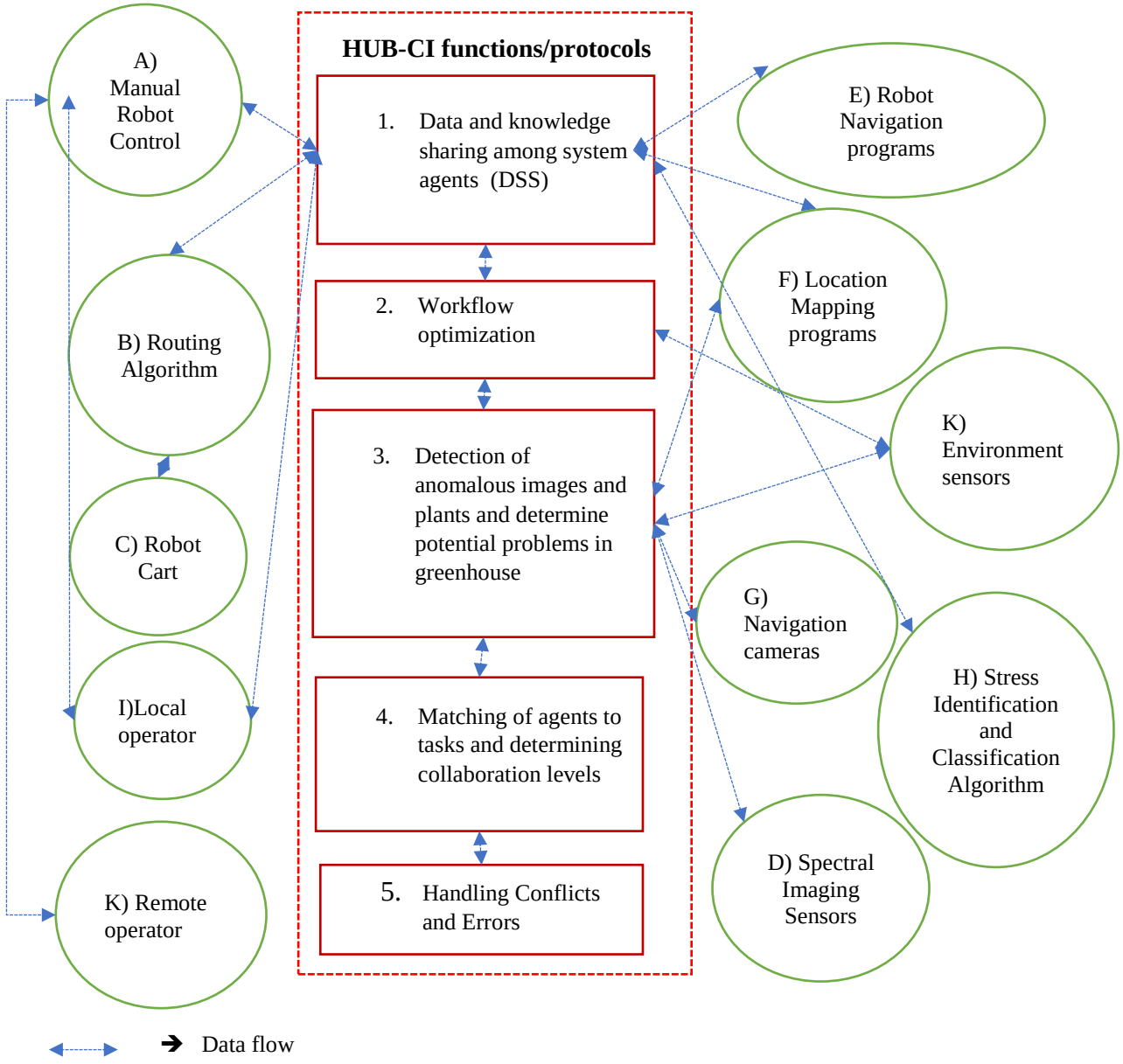


Figure 3-2 Diagram describing interaction of agents along with HUB-CI functions and processes

3.3 Unsupervised Learning methods – Clustering and Anomaly Detection

To build a HUB system in a precision agriculture domain, a much-needed feature and mechanism is a Decision Support System (DSS) (Mehta et. al, 2015). Unsupervised learning involves machine learning algorithms and processes that generates patterns and learns from data that has not been labelled, classified or categorized. The purpose of unsupervised learning is to extract valuable concepts or information from the data. The main tasks of unsupervised learning often include several clustering approaches and anomaly detection (also known as novelty detection, or outlier detection) (Wang et. al 2016). Two unsupervised machine learning algorithms have been included to enable this DSS for the agricultural robotic system namely k-means clustering and anomaly detection. One important purpose of the DSS is to find previously unknown patterns in the agricultural spectral images which could provide farmers, agricultural experts etc. better insights on the condition of the greenhouse system that were previously unknown. Hence k-means clustering, and spectral anomaly detection have been included as DSS tools.

3.4 HUB-CI function 1: Data and Knowledge sharing across the networked system

With regard to an agricultural greenhouse robotic system, the following functionalities are proposed:

- Decision Support System type: A combination of communication, data and knowledge driven database. System information and data: Time series information of the day to day operation of the system. Machine learning (Anomaly detection) of this data can detect existing and potential errors and conflicts.
- Domain knowledge: A repository of information on plant illnesses and stresses, agricultural best practices, case study information, plant information etc.

- System metadata: All knowledge (model, configuration, manuals etc.) of robots, sensors, operators etc.

3.5 HUB-CI function 2: Workflow Optimization- Ascertain task and data dependencies

Assumption: Data mining agents can provide required data to any agent in a timely manner.

Figure 3.3 is the workflow for the system and task and data dependencies for the task of detection of diseases in a greenhouse setting. It describes the Collaborative Control Theory (CCT) principle of E-Work Parallelism (described in Section 2.3) with regard to a robotic system for disease detection in a greenhouse. The tasks described in the figure are the key tasks identified for this system:

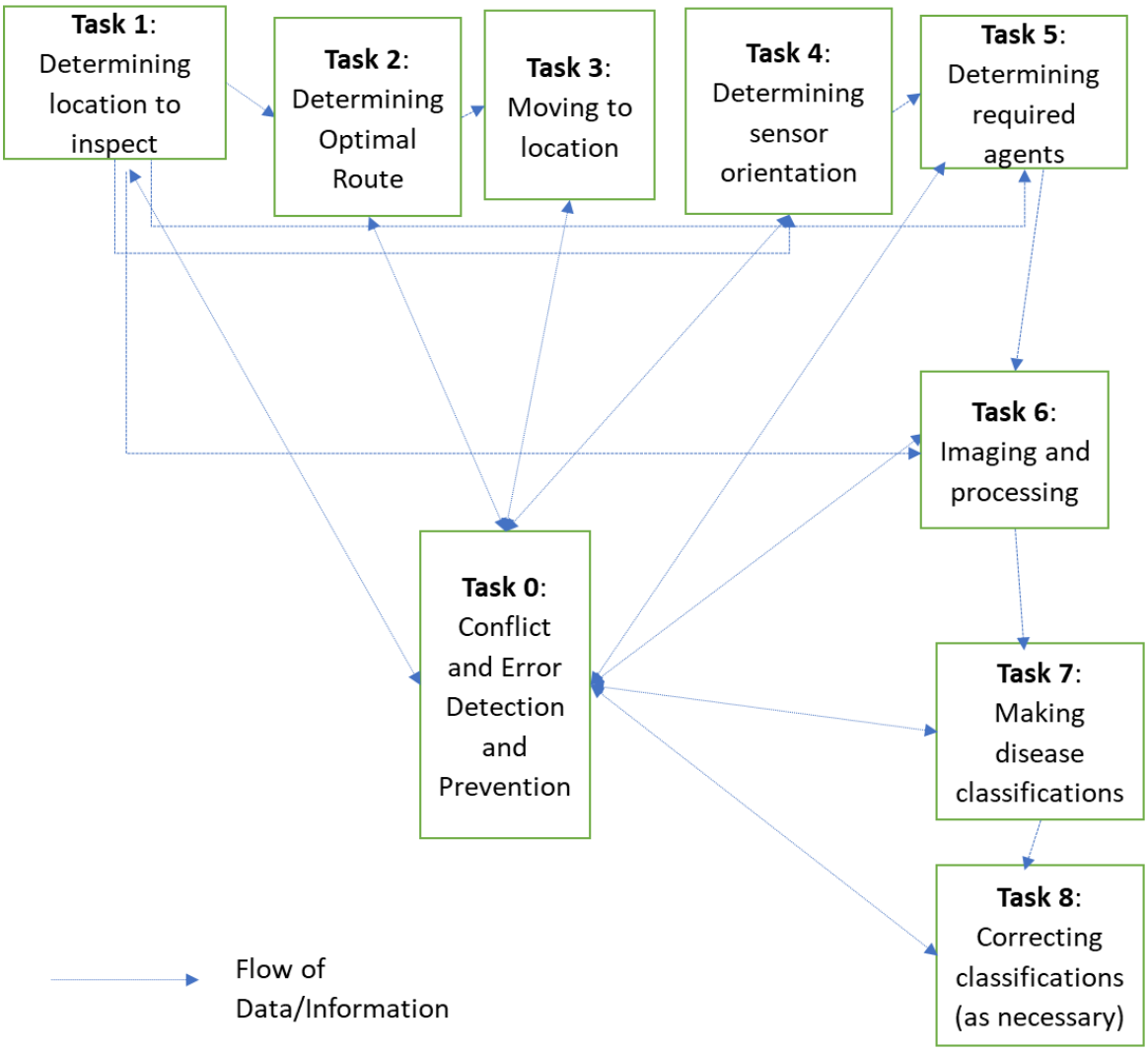


Figure 3-3 Key tasks for the networked agricultural robotic system and the corresponding flow of data/information

Based on the data dependencies identified in the Figure 3.3, Figure 3.4 is an example more optimized workflow. The optimized workflow in an implementation is intended to be collaboratively decided by decision making agents within the HUB depending on the specifics of the situation.

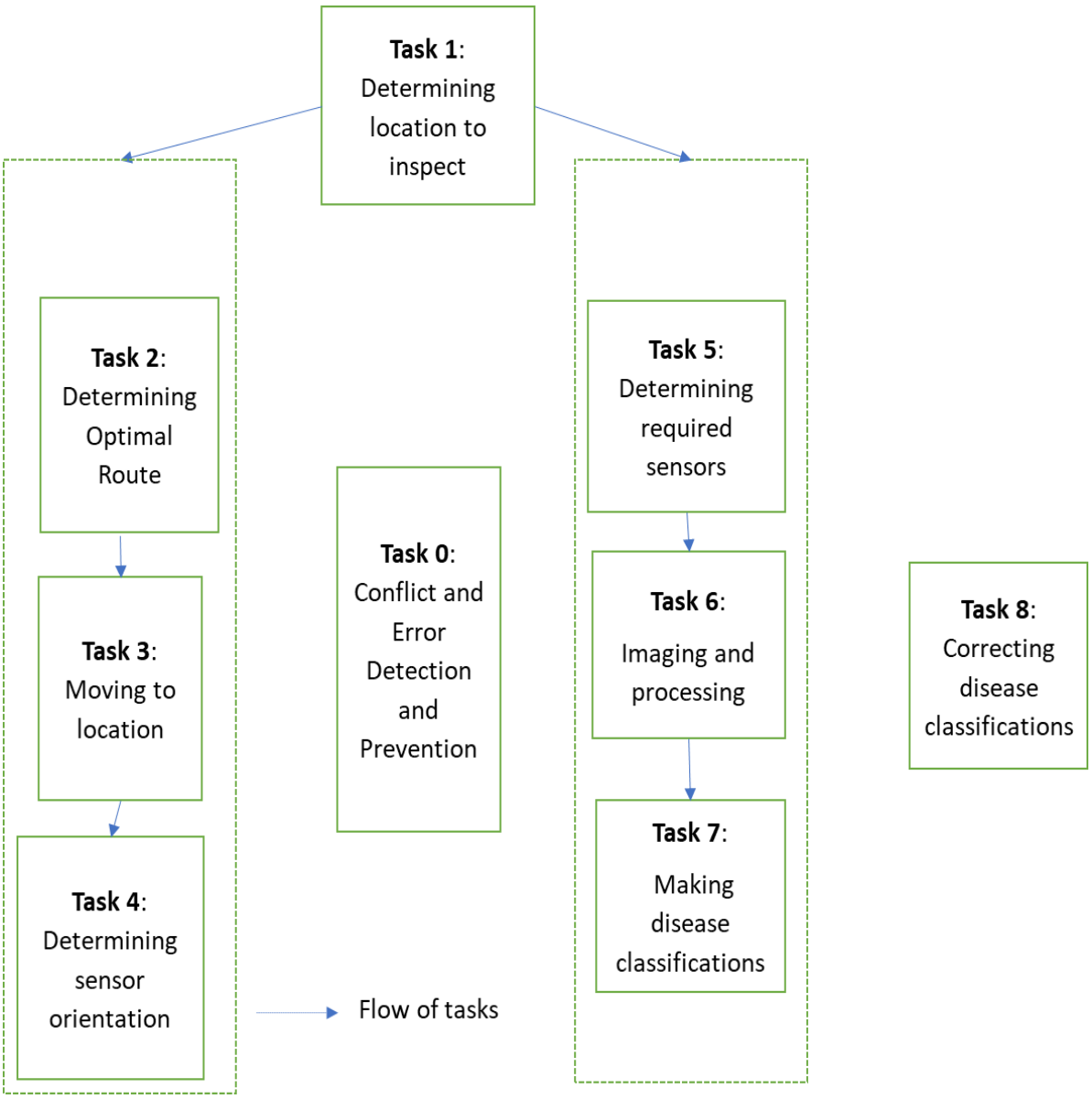


Figure 3-4 Example of an optimized and parallel workflow for semi-automated detection of diseases in a greenhouse

The characterization of system tasks and agents modelled here also highlights another relevant Collaborative Control Theory (CCT) principle i.e. ***Association and Dissociation (AD)***, described in section 2.3. Here is how the AD principle is in effect in this agricultural robotic system:

- Not all sensors need to participate in the detection and diagnosis process at a time.

  Example: Suppose out of n sensors, by using information provided by the DSS, if it is concluded that 2 specific sensors are optimal for the diagnosis of plant *i*, then the set of participating sensors S becomes S = $\{s_1, s_2\}$.

- Not all human operators are required to participate in system activities at all times. The system can calculate service level (SL) for each of the operators and pick the most required operators based on the problem.

- Service levels (SL) of the agent can be compared relative to the collaborative network. SL can be calculated by the following logic based on that proposed in Nof et al., 2015:

  $$SL_{\omega_a} = Pr(D_{\omega_a} < K_{\omega_a}) \rightarrow SL_{\alpha} = Pr(D_{\alpha} < K_{\alpha}) \quad (1)$$

  $D_\omega \rightarrow$ Demand of agent, $K_\omega \rightarrow$ Capacity of agent, $D_\alpha \rightarrow$ Demand of collaborative network, $K_\alpha \rightarrow$ Capacity of collaborative network

3.6 HUB-CI function 3: Collaborative detection of anomalies in plants from spectral images

Research Question 3: What DSS tools are necessary for a) optimal collaboration with minimal error in an agricultural setting, and b) enable early detection of stresses in plants?

Figure 3-5 presents an overview of the protocol for human robot collaborative detection of anomalies in plants.

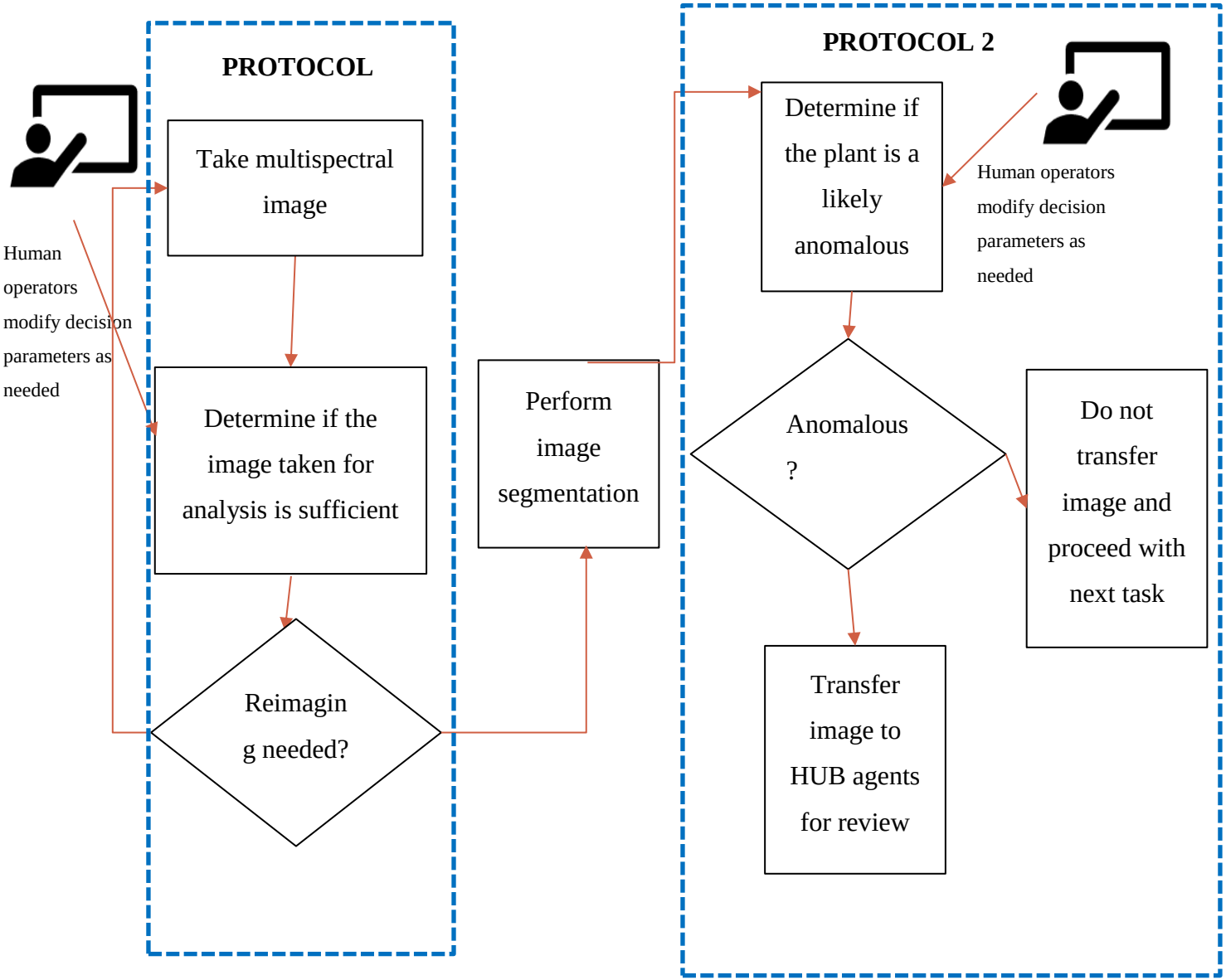


Figure 3-5 Protocol for detecting anomalous plants in a semi-automated greenhouse

3.6.1 Protocol 1: Detecting anomalous images captured and performing segmentation

This protocol can be summarized in the following two questions:

1. How can the ARS HUB-CI system determine automatically if the image taken may not be sufficient for analysis or if an operator needs to review the plant? Example image anomalies include but are not limited to:
    - Weeds or grasses in the background
    - Other aspects in the foreground that compromise precise imaging of the leaves
    - Poor lighting
2. How to determine which part of the image to extract for analysis i.e. how to segment out the leaf?

Solution: Use k-means clustering to create a predetermined number of clusters for each image. If the correct bands and number of clusters are used the leaf will be a distinct cluster. If the number of pixels for the leaf cluster do not meet minimal thresholds, then the imaging must be performed again. Figure 3.6 describes the process methodology for imaging plants in order to determine their condition.

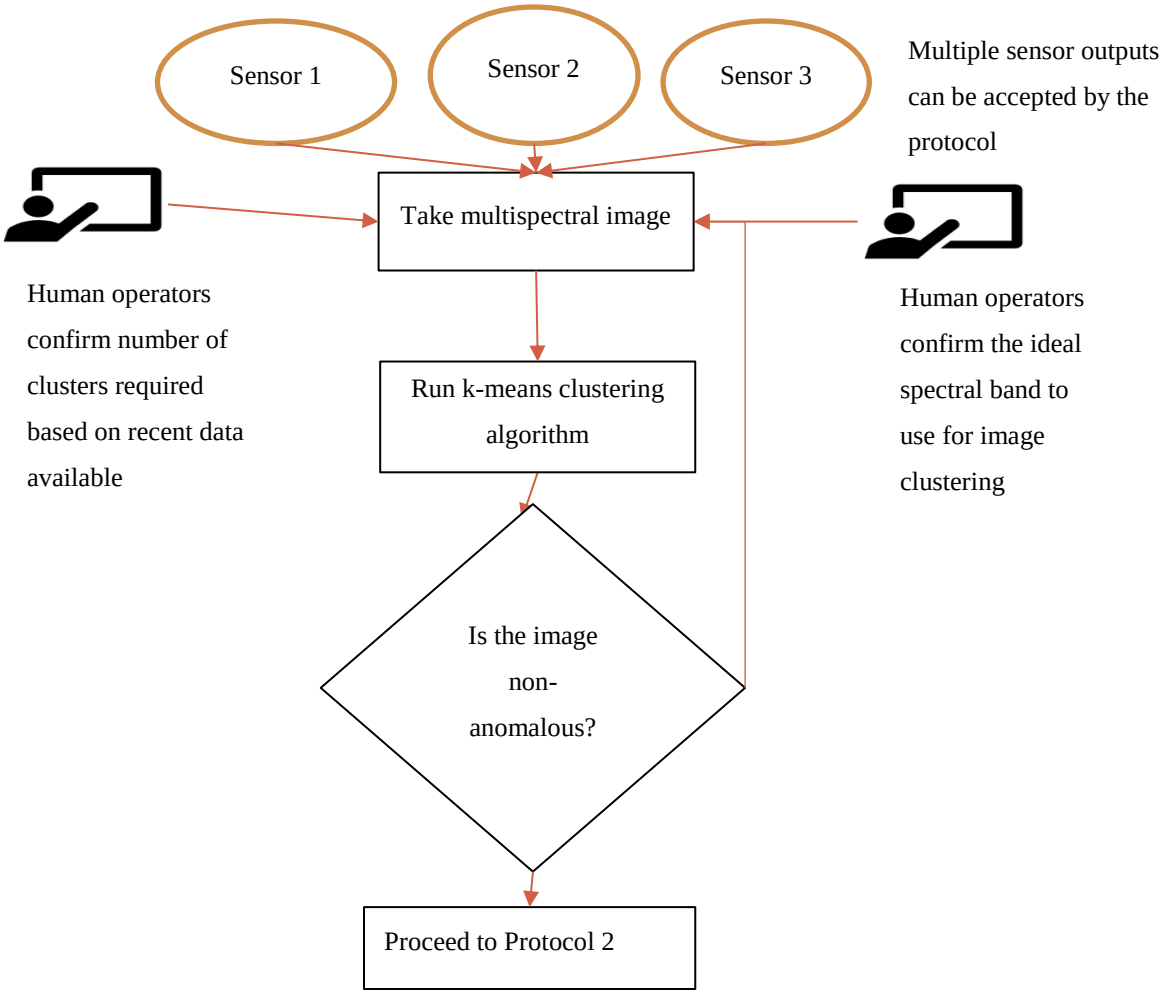


Figure 3-6 Protocol for collaborative imaging of plants

3.6.2 Protocol 2: Detecting anomalies in plants

Problem description:

- All images collected does not need to go on the HUB and sent to remote operators.
- In monitoring greenhouse plants, an anomaly may be detected in the plant but cannot yet identify or classify it to a specific stress. Another type of anomaly could be when there is not enough quality data to make a classification or when the symptoms of different stresses are quite similar.

- Most importantly, how to enable early detection of anomalous pepper plants likely to have biotic or abiotic stress?

Solution: Use anomaly detection on the spectral images. Statistical anomaly detection on the spectral images

Description: The protocol uses the squared Mahalanobis distance as a measure of how anomalous a pixel is with respect to an assumed background.

$$D_M(\vec{x}) = \sqrt{(\vec{x}-\vec{\mu})^T S^{-1}(\vec{x}-\vec{\mu})} \quad (2)$$

$x \rightarrow pixel\ spectrum\ , \mu \rightarrow background\ mean,\ \ S^{-1} \rightarrow background\ covariance$

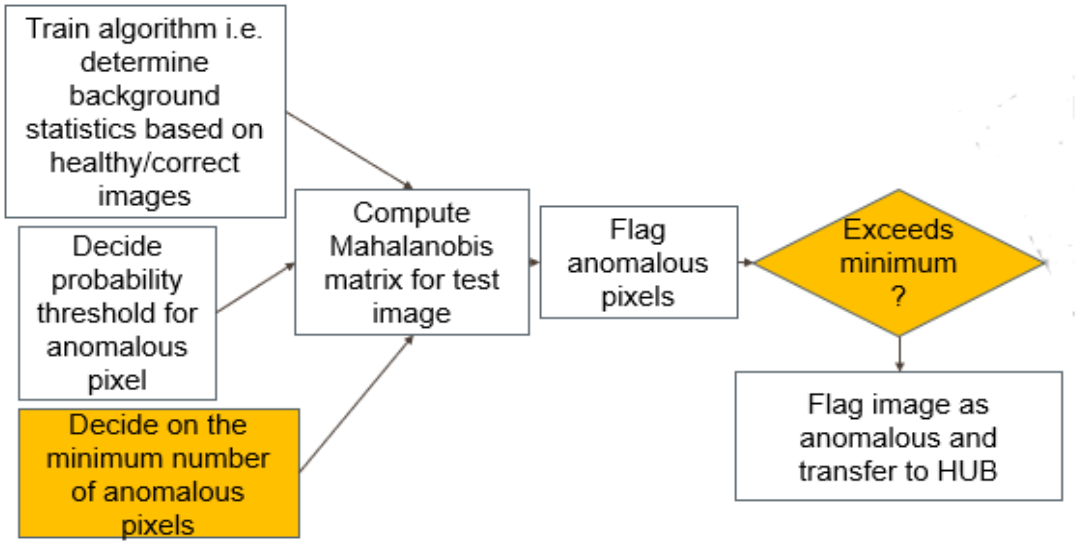


Figure 3-7 Protocol for collaborative detection of plant anomalies

To declare pixels as anomalous, a threshold Mahalanobis distance score must be specified. One method is to choose all image pixels whose score has say a probability of less than 0.01, for example. The chi-squared distribution was used to do this. The inputs to the chi-squared quantile function include desired probability and degrees of freedom (df).

$$f(x, df) = \frac{x^{df-1}}{2^{(\frac{df}{2}-1)}\gamma(\frac{df}{2})} \quad (3)$$

$x \rightarrow specific\ pixel\ , \gamma \rightarrow\ gamma\ function,\ \ df \rightarrow degrees\ of\ freedom$

3.7 HUB-CI function 4: Best Matching of Networked System Agents to Tasks

3.7.1 Collaboration Requirement Planning (CRP) and Best Matching (BM)

The CCT principle of Collaboration Requirement Planning (CRP) (section 2.3) is relevant here.

- Set S denotes the *combination of agents* required to resolve issue D
- S = {s_1, s_2, s_3, s_4, s_5} and D = {d_1, d_2, d_3, d_4}
- Then the Collaboration Requirement Matrix (CRM) and for would look similar to:

$$\text{CRM} = \begin{bmatrix} 1 & 2 & 0 & 1 \\ 0 & 1 & 3 & 2 \\ 1 & 1 & 1 & 0 \\ 0 & 3 & 2 & 1 \\ 2 & 0 & 1 & 1 \end{bmatrix}$$

- The value range [0,3] indicates an increasing "Preference."

Furthermore, the CCT principle of Best Matching (section 2.3) is most applicable for this system to match individual or groups of agents to specific tasks i.e. issues detected in plants. A Best Matching formulation can be set up as follows:

- I {Agents} → J {Problem Areas in the greenhouse(s)}
    - Multiple agents (≥ 2) can be matched to a single problem area at a time
    - P_{ij} is based on the Preference matrix calculated on basis of available sensors and agents required at area j
    - Robot availability is a key constraint
    - Optimizes the detection of issues based on sensor requirements and availability

The assignment problem formulation can hence be written as:

$$Maximize \sum_{\in I} \sum_{\in J} P_{ij} * x_{ij} \quad (4)$$

s.t,

$$\sum_{\in I} x_{ij} \leq 2, \qquad i \in I \quad (5)$$

$$\sum_{\in J} x_{ij} = 1, \qquad j \in J \quad (6)$$

$$x_{ij} = \{0, 1\}, \quad i \in I, j \in J$$

3.7.2 Collaboration Strategy for Collaborative Monitoring of Greenhouse Plants

Collaboration Strategy: Human operator performs supervisory tasks and tasks requiring superior expertise, perception and skill, whereas robot performs manual and repetitive tasks.

Table 3.2: ARS System without HUB-CI vs ARS System with HUB-CI

Without HUB-CI	With a HUB-CI based Decision Support
1) Robot checks all plants on each and every run 2) Assuming a supervisory role Human operator has to verify all the diseased or anomalous cases 3) Human operator will also be required to verify a few randomly sampled cases that were detected as healthy	1) The HUB-CI will use a) Information contained in one or several knowledgebases, b) inputs and outputs of multiple agents (human or automated), c) Outputs of HUB-CI based tools for Decision Support (described in section 3), to generate probabilities of the condition of the plant (diseased, healthy, low N_2 content etc.) in several areas of the greenhouse. 2) Each and every plant in the greenhouse does not need to be monitored at each monitoring cycle. The integrated planner based on the HUB-CI assigns areas in the greenhouse which need to be monitored and by which agent (robot or human). 3) Human operator checks certain areas for anomalies that require some domain knowledge and expertise, and which cannot be automatically detected and determined by an algorithm. 4) The robot with mounted detection sensors CONCURRENTLY checks areas which are most likely to contain diseased plants. These diseases can be detected by a corresponding disease detection algorithm. 5) After both agents (human and robot) have concurrently performed tasks that have been assigned based on their strengths, the human operator in a supervisory role may random sample some of the detections performed by the robot to ensure accuracy. 6) After this task both agents should random sample a few plants in the areas that are determined to have a low probability for diseased and other anomalous plant cases to ensure that the plants are in a healthy condition.

Assumptions

1) Diseases in plants will tend to spread to other plants that are nearby. Certain areas of the greenhouse are expected to have certain diseases and defects
2) Equipment inspection, servicing and maintenance is a fixed cost.

3) The human operator/agent involved has sufficient domain knowledge in agriculture and has working knowledge regarding management of greenhouse crops.
4) Owing to a) a superior domain knowledge and expertise with regard to agricultural aspects, and b) better perception and adaptability to surroundings/conditions; the human operator/agent involved can identify and categorize the various other plant anomalies that are not included in the automated disease detection algorithm.
5) The HUB-CI based integrated planner and DSS is periodically updating based on new inputs and information, local and global from multiple agents and knowledgebases.
6) Human operator takes the same amount of time to move between two locations of a greenhouse. Time to move by the human operator is marked as fixed and is relatively insignificant for this problem as he may begin from any location in the greenhouse and can move without any encumbrance that a robot cart would have.

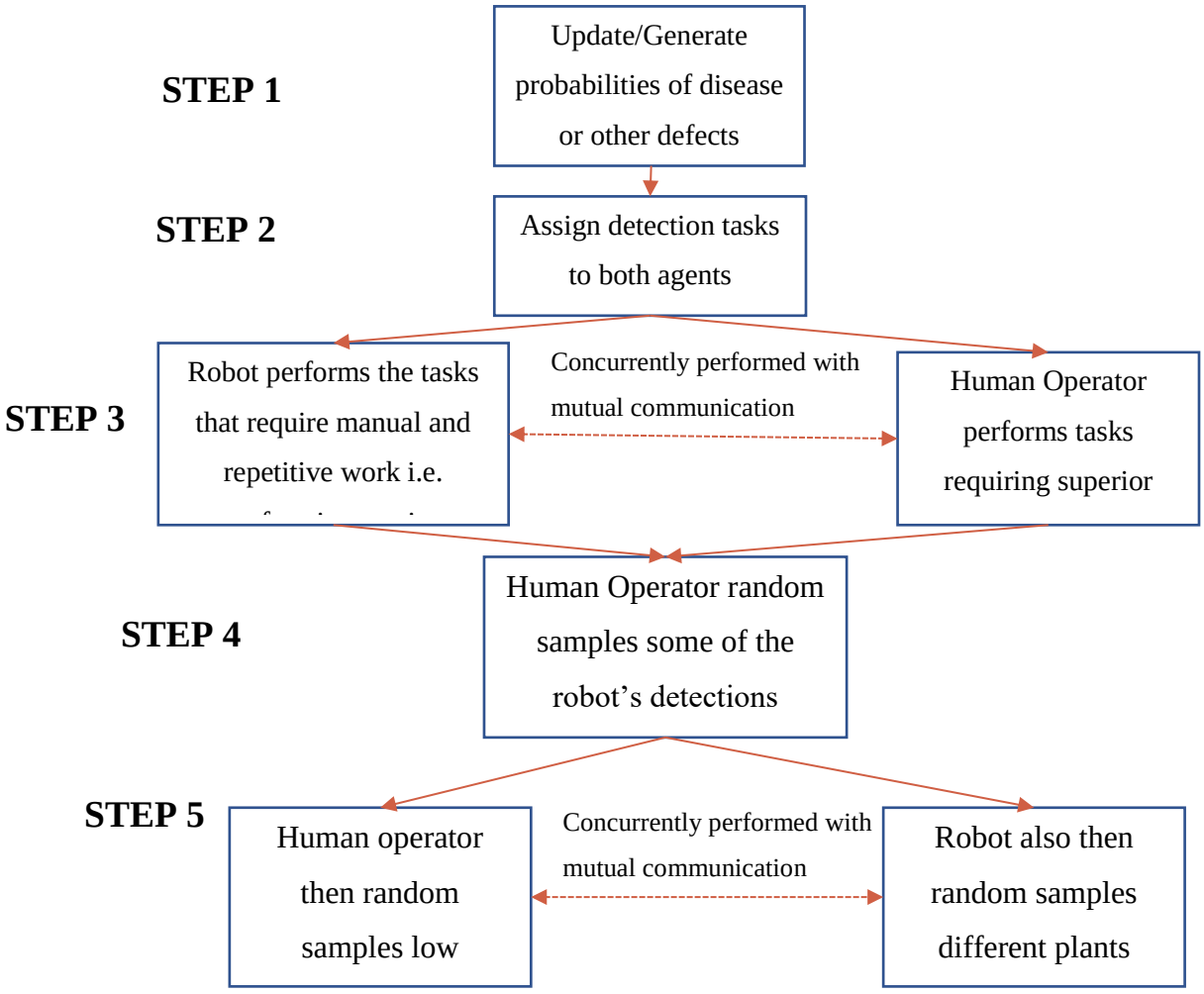


Figure 3-8 Workflow diagram for HUB-CI Collaboration Strategy

3.8 HUB-CI function 5: Handling Conflicts and Errors

Objective: Maximize resolution of Conflicts and Errors (C&Es)

Conflicts and Errors are to be expected in an Agricultural Robotic system given the relatively unstructured nature of agriculture. The goal of the HUB-CI based networked telerobotics is to implement a collaborative process for disease detection with enhanced decision-making tools and processes. Below are *some expected errors and conflicts*:

Table 3.3: List of potential Errors and Conflicts

Type	**Error or Conflict name**	**Description**	**Agents concerned**	**Agents and Information required for resolution**
Error	Routing and navigation error	• Robot cart cannot follow prescribed routing plan • System unable to generate a routing plan	• Automated Routing algorithm • Navigation agent • Robot cart • Human agent	Human agent with navigation sensor data
Error	Imaging anomalies	• Imaging sensors do not capture the desired plant part	• Spectral image sensors • Image segmentation and anomaly detection agent	Human agent with data from Spectral sensors and outputs from image segmentation agent.
Conflict	Conflict between Routing agent, imaging agents, and stress detection agents	Contradictory commands for robot cart from imaging, stress detection and routing agents.	• Robot cart • Image segmentation agent • Anomaly and stress detection agent • Routing algorithm • Navigation agents	HUB-CI protocol to manage interaction between automated agents

Table 3.3 continued

Conflict	Plant diagnosis conflict	• Disagreements between one or more human agents regarding nature of plant stress • Disagreement between one or more human agents regarding existence of stress in specific plant	Multiple human operators across the network. Examples include Agricultural experts, farmers, system engineers etc.	Human agent like the system supervisor and the parties in conflict.
Conflict	Procedural conflict	Disagreements between one or more human agent regarding system procedures and processes for disease detection	Multiple human operators across the network. Examples include Agricultural experts, farmers, system engineers etc.	Human agent like the system supervisor and the parties in conflict.
Error	Navigation sensor failure	Faulty navigation sensor outputs.	• Navigation sensor(s) • Mapping agents • Navigation agents	Trained human operator with knowledge of navigation sensors/cameras

Assumptions

1. Static/well-defined conflict and error classifications
2. Broad categories for conflict and error classifications due to absence of prior knowledge
3. As there is no a priori knowledge, the list of conflicts and errors described above are not exhaustive and hence new categories of conflicts and errors can be defined. Furthermore, existing error and conflict categories may also be modified in future as needed.

Formalization of Objective

- Maximize Resolution of Errors and Conflicts

$$\text{Errors and Conflicts resolved} = w_1 \sum \alpha_1 + w_2 \sum \alpha_2 + w_3 \sum \alpha_3 + w_4 \sum \alpha_4 + w_5 \sum \alpha_5 + \ldots\ldots + w_n \sum \alpha_n \quad (7)$$

Subject to the following ***time constraint(s)***:

- $\frac{\sum t_{\alpha_m}}{\sum \alpha_m} \leq s\ minutes$ (Average time to resolve conflict/error α in category *m* of Conflict/Error cannot exceed *s* minutes) (8)

Where,

- w_k are constants represent the relative importance of the category *m* of Conflict/Error (as defined in the Table 3.2).
- α_m represents the conflict/error α in category *m* of Conflict/Error (as defined in Table 3.2)

4. EXPERIMENTS AND RESULTS

"HUB-CI Function 3" and the Collaboration Strategy for Collaborative Monitoring of Greenhouse Plants from Section 3.7.2 have been tested and demonstrated in the experiments in this section. The other functions were described as part of the design of the HUB-CI DSS and to illustrate how it works for the purpose of agricultural greenhouse monitoring.

4.1 Simulation of Collaboration Strategy

Plants in this section as per HUB-CI are expected to be Healthy	**Plants in this section as per HUB-CI are expected to have Disease X**	**Plants in this section as per HUB-CI may require to be checked by expert**

Figure 4-1 Example of DSS outputs for Greenhouse sections using collaborative machine learning protocols

The workflow diagram described in Section 3.6.2 and the baseline scenario (no HUB-CI) were programmed using Python programming (2 programs), and the following experiments were conducted on it.

Experiment 1a): How much more efficient is a HUB-CI system with an Integrated Planner (based on DSS) compared to a system that does not harness collaborative intelligence?

Setup

- A plant can be either a) Healthy, b) With Disease, or c) Having other anomalies
- For the scenario without the HUB-CI system, the plants "With Disease," "Healthy" and "Having Other anomalies" are randomly determined using Python Numpy package's "random.randint".

- The task requiring superior perception and skills that for the human operator to perform are set as the plants of the condition “Having other anomalies.”
- For this scenario with the HUB-CI system the plants “With Disease,” “Healthy” and “Having Other defects” are allotted specific probabilities at different zones of the greenhouse
- It takes 3 minutes for the robot to a perform detection task, which involves moving to the adjacent plant, and performing a detection/evaluation of the plant.
- It takes a human operator the same time as the robot i.e. 3 minutes to perform a detection task which involves moving to the plant and performing a detection/evaluation of the plant.
- In Step 4, the human operator randomly checks 10% of the detections performed by the robot.
- In Step 5, the robot and the human operator EACH sample 20% of the plants expected to be healthy. As these 2 agents are communicating via the HUB, these 2 agents *do not sample the same plants*.
- 20 runs of the simulation for both the systems were performed

Table 4.1: Variables for experiment 1a)

Controlled variables	**Dependent variables**
1) Time for robot to perform a detection task = 3 minutes 2) Time for human operator to perform a detection task = 3 minutes 3) The number of healthy, diseased, or "other anomalous" cases are uniformly distributed across all plants in the greenhouse. 4) Step 4: Human operator randomly checks 10% of the robot's detections 5) Step 5: Human and robot both sample 20% of the plants expected to be healthy 6) Total number of plants in greenhouse = 400	1) Time to complete (TTC) 2) Defects detected (D)

Performance metric:

$$Efficiency\ ratio\ (\eta) = \frac{Defects\ detected\ (D)}{Time\ to\ Complete\ (TTC)} \quad (9)$$

This efficiency ratio was chosen as it is important to optimize the defects detected and also minimize the time required to perform routine inspections. The underlying assumption is that a higher time to complete implies an increase in cost of operation for the overall system which would include human operator costs, energy costs etc.

Table 4.2: Summary of Results for Experiment 1a)

	Average Defects Detected	Average TTC (minutes)	Standard deviation (Defects)	Standard Deviation (TTC)	η
Without HUB-CI	262.65	2017.95	10.39	31.18	0.13
With HUB-CI	240.65	763.95	4.37	12.77	0.32

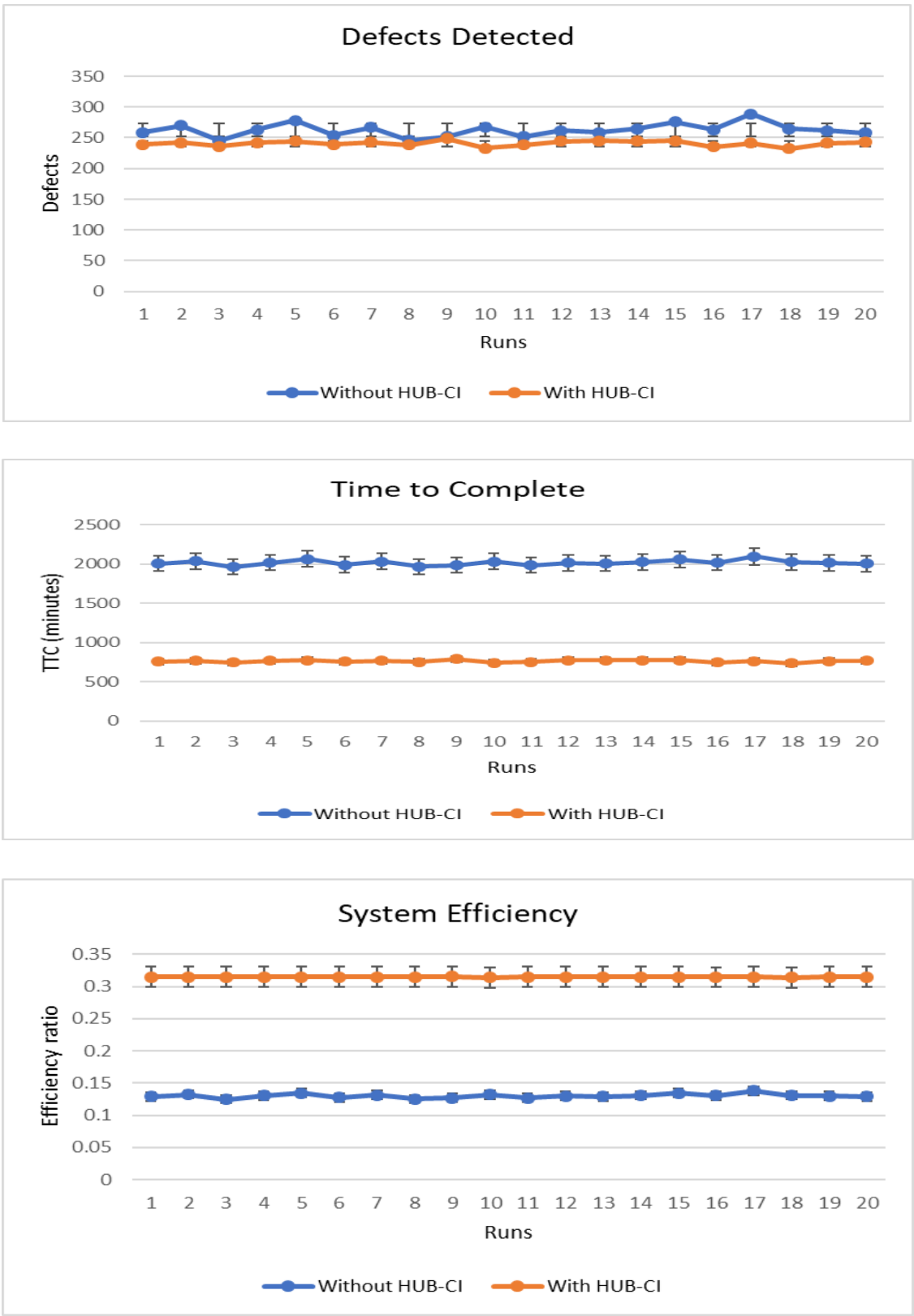


Figure 4-2 Performance of the system with HUB-CI vs a system without HUB-CI

Table 4.3: Statistical significance for experiment 1a)

Two-sided t-test (95% Confidence Interval)			
	T-Value	P-Value	DF
Defects Detected	8.73	0	25
TTC	166.43	0	25
η	-271.22	0	19

Observations

The preliminary run indicates that this system with HUB-CI is expected to have a better Efficiency and take only 38% of the time taken by a system that has no HUB-CI which is significant as it demonstrates the superior efficiency that comes with planned human robot collaboration as opposed to unplanned collaboration or agricultural monitoring systems that are purely manual or purely automated. The defects detected may be slightly lower, but this can be accounted to the decision to have the human and robot sample only 20% of the plants expected to be healthy in step 5. This will be addressed in subsequent sections.

Experiment 1 b): What would be the HUB-CI system performance be if the tasks performed by the human operator i.e. the tasks requiring superior perception and skills are expected to take more time than the tasks performed by the robot i.e. tasks that require manual and repetitive work?

Setup

- Most conditions are set the same as experiment 1a) with the following considerations:
  - It takes 3 minutes for the robot to a perform detection task, which involves moving to the adjacent plant, and performing a detection/evaluation of the plant.
  - It takes a human operator ***twice the time as the robot*** that is 6 minutes to perform a detection task which involves moving to the plant and performing a detection/evaluation of the plant. This adjustment has been made to both systems.

- When it comes to the supervisory task of checking the robot's detection, the human operator will still take 3 minutes as in Experiment 1a)

Table 4.4: Variables for experiment 1b)

Controlled variables	Dependent variables
1) Time for robot to perform a detection task = 3 minutes 2) Time for human operator to perform a detection/evaluation task = 6 minutes 3) Time for the human operator to perform a supervisory task = 3 minutes. 4) The number of healthy, diseased, or "other anomalous" cases are uniformly distributed across all plants in the greenhouse. 5) Step 4: Human operator randomly checks 10% of the robot's detections 6) Step 5: Human and robot both sample 20% of the plants expected to be healthy 7) Total number of plants in greenhouse = 400	1) Time to complete (TTC) 2) Defects detected (D)

Table 4.5: Summary of Results for Experiment 1b)

	Average Defects Detected	Average TTC (minutes)	Standard deviation (Defects)	Standard Deviation (TTC)	η
Without HUB-CI	268.8	2842.8	11.21	67.29	0.09
With HUB-CI	240.65	1068.45	4.92	19.72	0.23

Table 4.6: Statistical significance for Experiment 1b)

Two-sided t-test (95% Confidence Interval)			
	T-Value	P-Value	DF
Defects Detected	10.28	0	26
TTC	113.17	0	22
η	-293.64	0	34

Observations

Despite doubling the expected task time for the human operators, the average Time to Complete of the system with HUB-CI is still less than half of that for the system without HUB-CI. The number of defects detected is lower on average but that can be attributed to the relatively low number of plants sampled in Step 5, which has been corrected in the next experiment run. System efficiency is more than double than that of the case where there is no HUB-CI.

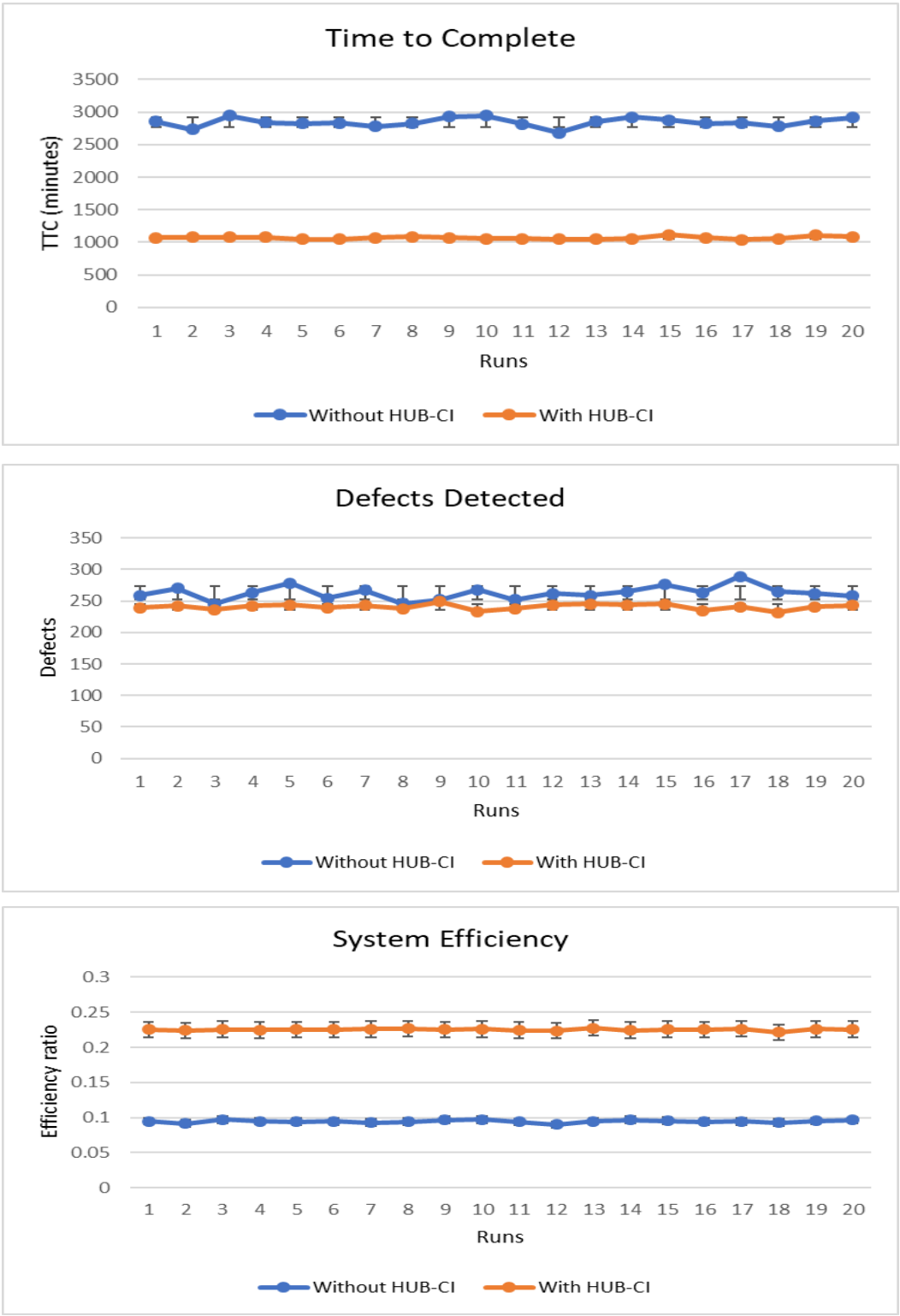


Figure 4-3 Performance of the system with HUB-CI vs a system without HUB-CI

Experiment 1 c) Assuming the conditions of experiment 1 b), how can the number of detections made by the HUB-CI integrated planner equal or exceed those made by the baseline system and yet have a better TTC compared to the baseline scenario?

- Most conditions are set the same as experiment 1b) i.e.:
  - It takes 3 minutes for the robot to a perform detection task, which involves moving to the adjacent plant, and performing a detection/evaluation of the plant.
  - It takes a human operator ***twice the time as the robot*** i.e. 6 minutes to perform a detection task which involves moving to the plant and performing a detection/evaluation of the plant. This adjustment has been made to both systems.
  - When it comes to the supervisory task of checking the robot's detection, the human operator will still take 3 minutes as in Experiment 1a)
- The major change made here is:
  - In step 5, human and robot both sample 45% of the plants expected to be healthy (more than double the amount sampled in Step 5 for experiments 1a) and 1b) but still lesser than the total number of plants sampled under the baseline (no HUB-CI) scenario).

Table 4.7: Variables for experiment 1c)

Controlled variables	**Dependent variables**
1) Time for robot to perform a detection task = 3 minutes 2) Time for human operator to perform a detection/evaluation task = 6 minutes Time for the human operator to perform a supervisory task = 3 minutes. 3) The number of healthy, diseased, or "other anomalous" cases are uniformly distributed across all plants in the greenhouse. 4) Step 4: Human operator randomly checks 10% of the robot's detections 5) Step 5: Human and robot both sample 45% of the plants expected to be healthy 6) Total number of plants in greenhouse = 400	1) Time to complete (TTC) 2) Defects detected (D)

Table 4.8: Summary of Results for Experiment 1c)

	Average Defects Detected	Average TTC (minutes)	Standard deviation (Defects)	Standard Deviation (TTC)	η
Without HUB-CI	270.45	2852.7	9.71	58.28	0.09
With HUB-CI	270.65	1202.4	4.57	21.87	0.23

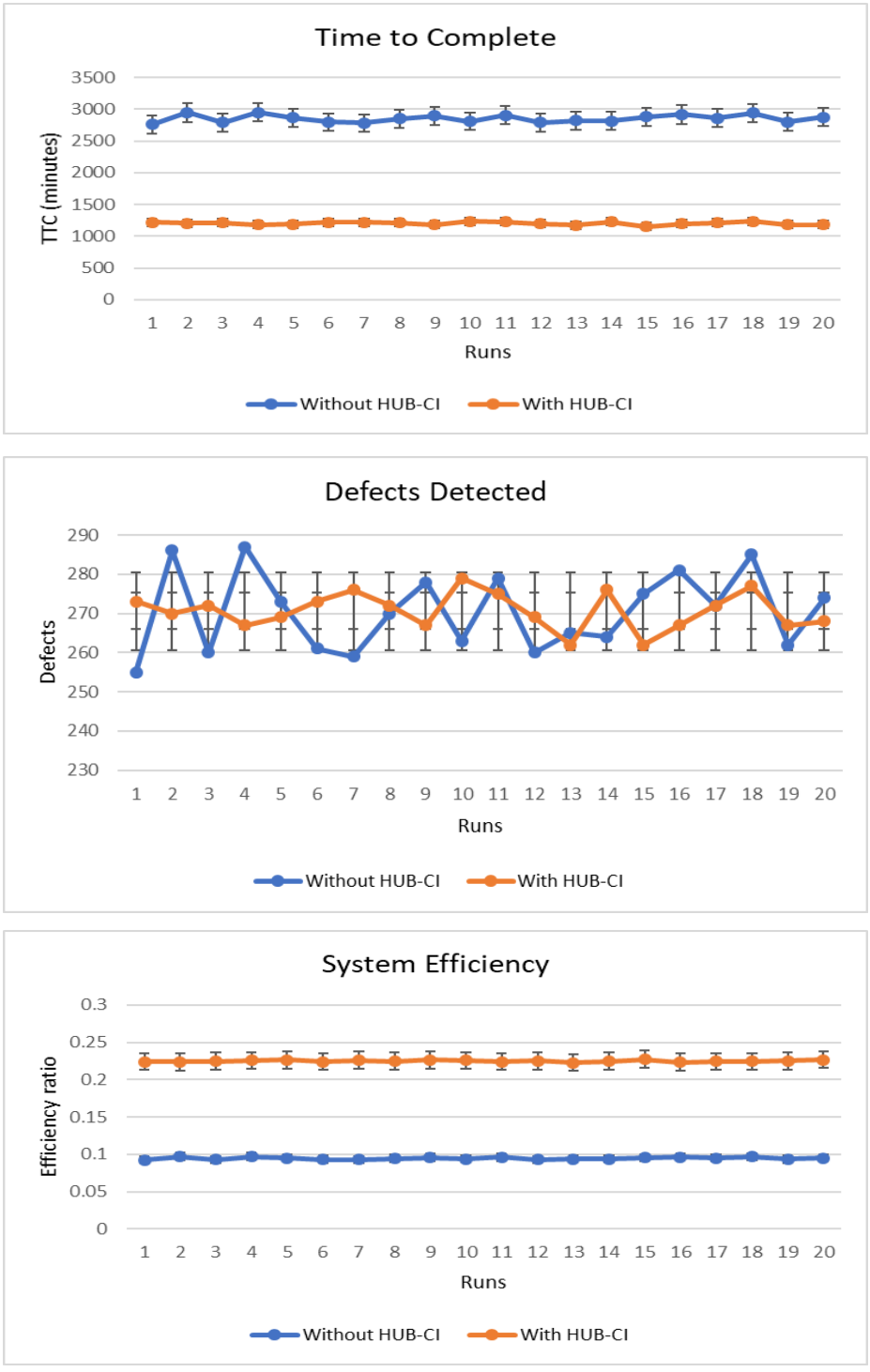


Figure 4-4 Performance of the system with HUB-CI vs a system without HUB-CI

Observations

As shown in the above Table, the average defects detected by the HUB-CI system match the case of when there is no HUB-CI system. The TTC for the HUB-CI system is still less than half of that with no HUB-CI system which is highly desirable. Increasing the percentage of plants sampled in Step 5 optimizes the number of defects detected. System efficiency is once again superior for the monitoring system that uses HUB-CI.

Table 4.9: Statistical significance for Experiment 1c)

Two-sided t-test (95% Confidence Interval)			
	T-Value	P-Value	DF
Defects Detected	-0.08	0.934	27
TTC	118.56	0	24
η	-336.93	0	35

Experiment 1 d) Does increasing the number of plants impact the performance of the collaboration strategy using the HUB-CI system? Assuming all the other conditions of experiment 1 c), how will increasing the number of plants impact the performance of this system?

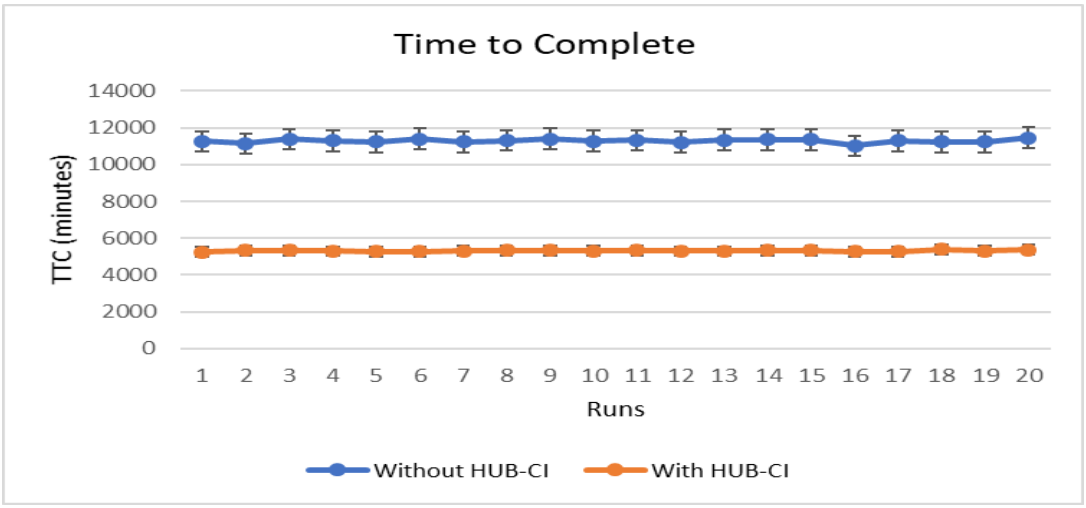
- Most conditions are set the same as experiment 1c) with the following change:
  - The number of plants in the greenhouse has been increased from 400 to 1600.

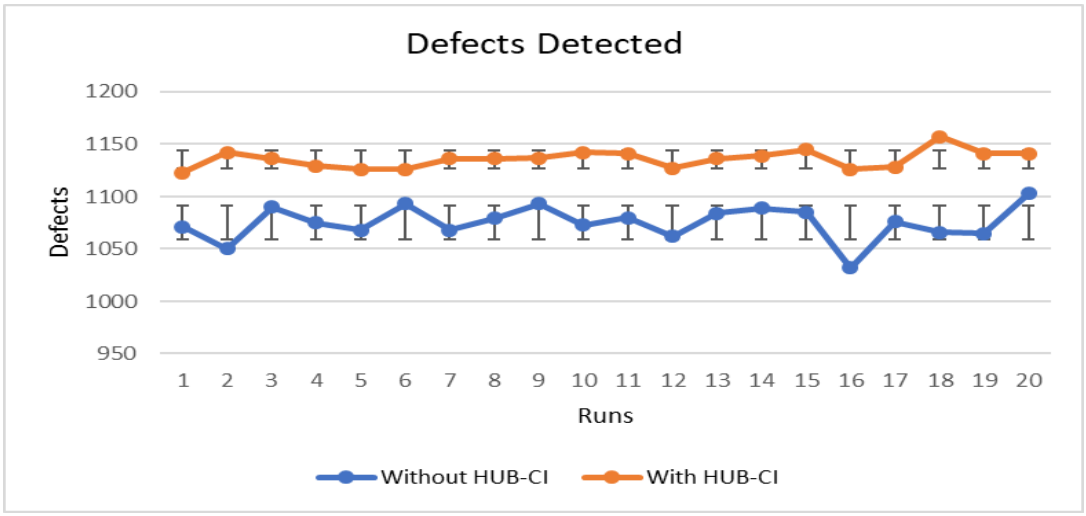
Table 4.10: Variables for Experiment 1d)

Controlled variables	**Dependent variables**
1) Time for robot to perform a detection task = 3 minutes 2) Time for human operator to perform a detection/evaluation task = 6 minutes Time for the human operator to perform a supervisory task = 3 minutes. 3) The number of healthy, diseased, or "other anomalous" cases are uniformly distributed across all plants in the greenhouse. 4) Step 4: Human operator randomly checks 10% of the robot's detections 5) Step 5: Human and robot both sample 45% of the plants expected to be healthy 6) Total number of plants in greenhouse = 1600	1) Time to complete (TTC) 2) Defects detected (D)

Table 4.11: Summary of Results for Experiment 1d)

	Average Defects Detected	Average TTC (minutes)	Standard deviation (Defects)	Standard Deviation (TTC)	η
Without HUB-CI	1075.1	11280.6	15.84	95.04	0.10
With HUB-CI	1135.7	5310.9	8.19	34.45	0.21





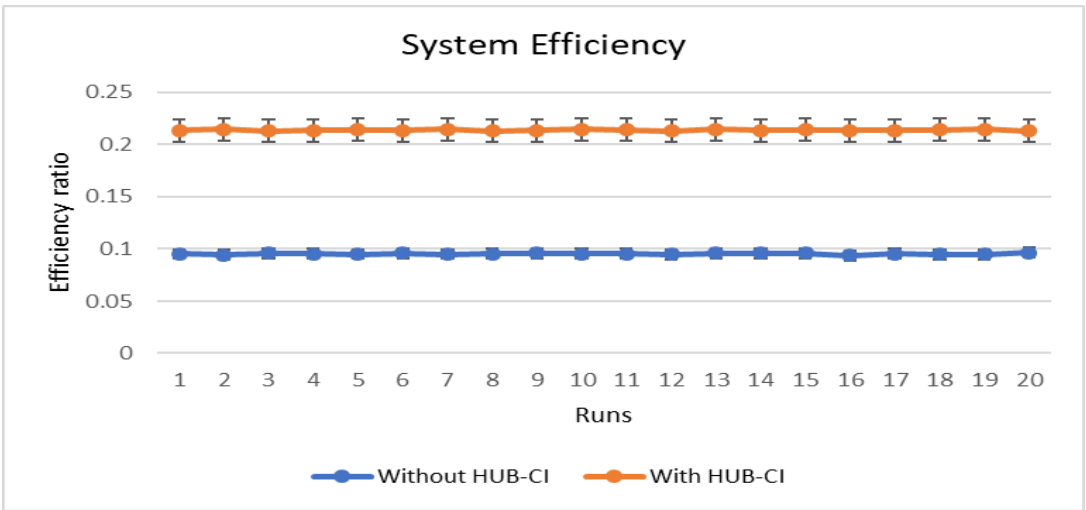


Figure 4-5 Performance of the system with HUB-CI vs a system without HUB-CI

Table 4.12: Statistical significance for Experiment 1d)

Two-sided t-test (95% Confidence Interval)			
	T-Value	P-Value	DF
Defects Detected	-15.2	0	28
TTC	264.08	0	23
η	-605.29	0	37

Observations

The HUB-CI system again delivers superior performance when compared to the baseline scenario on all metrics. An interesting outcome of this simulation shows that as the size of the system (number of plants) increase, the defects detected also improves. This fact is highly desirable as it indicates that in larger greenhouse systems, collaborative intelligence would be instrumental to system efficiency.

4.2 Protocol 1: K-means clustering analysis of the spectral images

Setup: Using trial and error, 50 spectral bands within the blue range were determined that would generate clusters that were unique to the leaf pixels i.e. those specific clusters did not merge with the background and was restricted to the plants. This step would make image segmentation feasible. A specific sensor or a multispectral camera could image the leaf using a specific band that would be optimal for image segmentation for the leaves.

Detecting anomalies in imaging: The k-means clustering was run on 30 hyperspectral images of pepper plants. 6 clusters were generated in 40 iterations. Detecting image anomalies via analyzing cluster centers vs. the hyperspectral band range can enable to automatically determine whether or not an image is likely anomalous (as defined in Section 3.5.1) and needs to be retaken. This was tested on the dataset with 30 images out of which 5 images were anomalous.

For the anomalous images the pattern of cluster centers vs the hyperspectral band range differed significantly from that of non-anomalous images.

Another key motivation behind using k-means clustering was for image segmentation. The k-means was run with 6 clusters generated over 40 iterations. Figure 3-6 are some examples of clusters used for image segmentation:

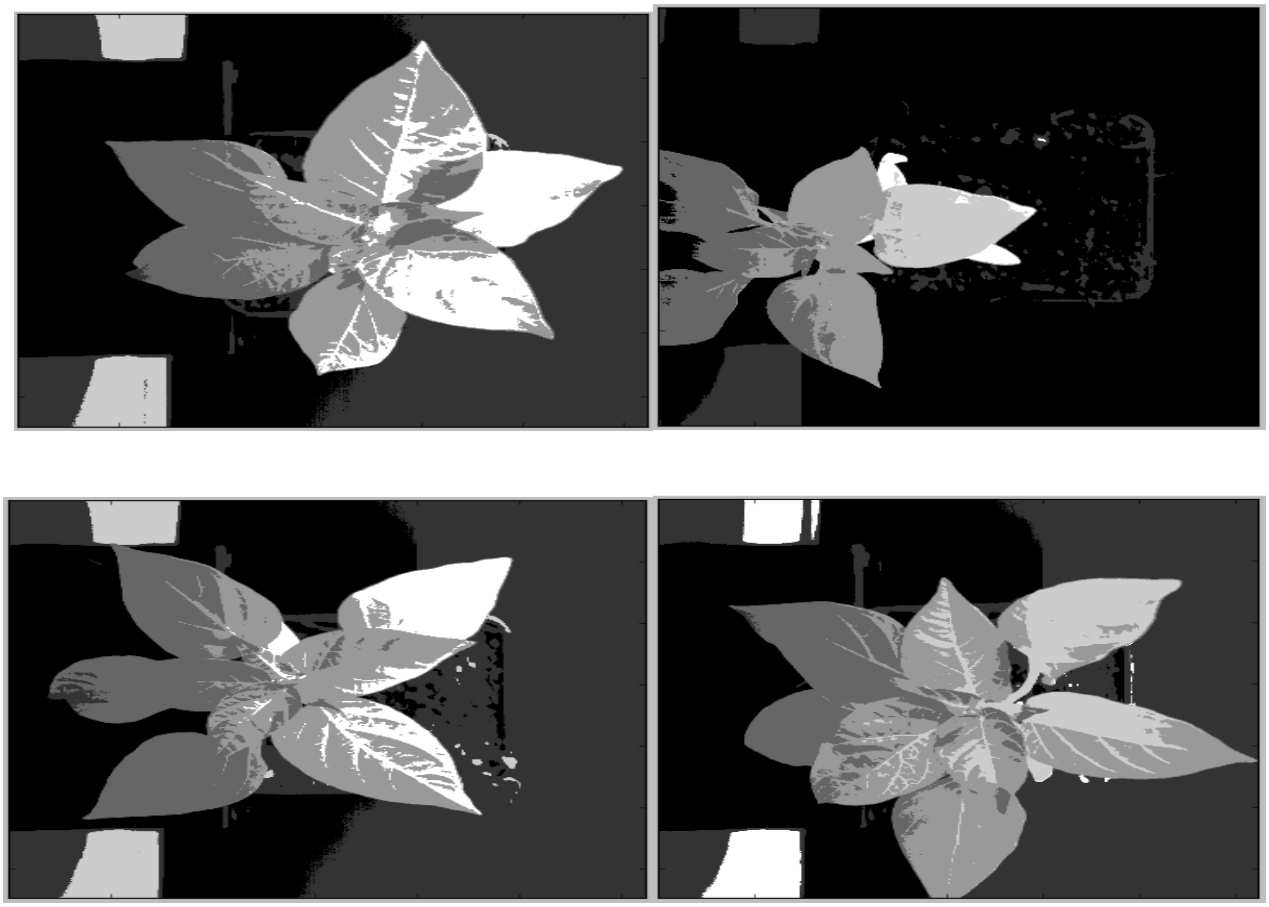


Figure 4-6 Generation of leaf clusters for image segmentation using k-means

Notice that in each of the above pictures, the clusters representing the leaves are different from those that represent objects in the background. This is where the choice of the spectral band range determined earlier in this section and the number of clusters that would enable the leaf

pixels to be differentiated from those in the background. At other spectral band ranges, clustering often resulted in large parts of the leaves merging with the background.

4.3 Protocol 2: Detecting anomalies in plants

Experiment 2:

Objective:

- To perform a check on Protocol 2 to evaluate its effectiveness with regard to detecting early anomalous plants that may have biotic or abiotic stresses.

Assumptions:

- Image segmentation has been performed with precision

Experiment setup and steps:

1. 28 uniformly distributed bands out of 840 spectral bands are selected for this analysis
2. The Gaussian statistics mean, covariance, number of samples (pixels) are calculated for each spectral band from the training sample which in this case was 14 spectral images of Healthy plants
3. The testing sample includes plants with 7 Tomato Spot Wilt Virus (TSWV), 7 Powdery Mildew (PM), 6 plants with both TSVW and PM, as well as 8 Healthy plants
4. The protocol for detecting anomalous plants described in Section 3.6.2 was run on the testing sample and for all images the Mahalanobis distance for each pixel was calculated
5. Pixels whose Mahalanobis distance values had a probability of less than 0.001 were identified by modeling the data with the chi-squared distribution using the number of bands sampled as the degrees of freedom (df). These pixels are considered anomalous.
6. Anomalous pixels that were not based on the plant images were removed manually

7. The mean, variance, and standard deviation of the number of anomalous pixels were considered for each image in the testing set.

Controlled variables: 1) Spectral bands selected, 2) Threshold probability to classify pixel as anomalous, 3) Degrees of freedom for the chi-squared distribution.

Dependent/test variable: Number of anomalous pixels

Table 4.13: Anomaly detection tests for Healthy plants, plants with TSVW and PM

Plant number	Class	Number of anomalous pixels based on threshold probability P < 0. 001
1	Healthy	10200
2	Healthy	30662
3	Healthy	15250
4	Healthy	5880
5	Healthy	9275
6	Healthy	7590
7	Healthy	27257
8	Healthy	87027
9	Powdery Mildew	59612
10	Powdery Mildew	71082
11	Powdery Mildew	68175
12	Powdery Mildew	61435
13	Powdery Mildew	54734
14	Powdery Mildew	86409
15	Powdery Mildew	79526
16	TSWV	207036
17	TSWV	219852
18	TSWV	55186
19	TSWV	206048
20	TSWV	157752
21	TSWV	201141
22	TSWV	326144
23	TSWV and PM	354830
24	TSWV and PM	326471
25	TSWV and PM	423569
26	TSWV and PM	407670
27	TSWV and PM	485720
28	TSWV and PM	532405

Table 4.14: Summary of Results for Experiment 2

	Mean	Variance	Standard Deviation
Healthy plants	24142.63	729811559	27015.03
TSWV plants	196165.57	6507810721	80671.00
PM plants	68710.43	127863478	11307.67
TSWV and PM plants	421777.5	6018922958	77581.72

Statistical significance test

A One-way ANOVA test was performed using Minitab 18 taking the four categories as factors.

Table 4.15: One-way ANOVA test summary

Source	DF	Adj SS	Adj MS	F-Value	P-Value
Factor	3	6.01824E+11	2.00608E+11	61.69	0.000
Error	23	74795173133	3251964049		
Total	26	6.76619E+11			

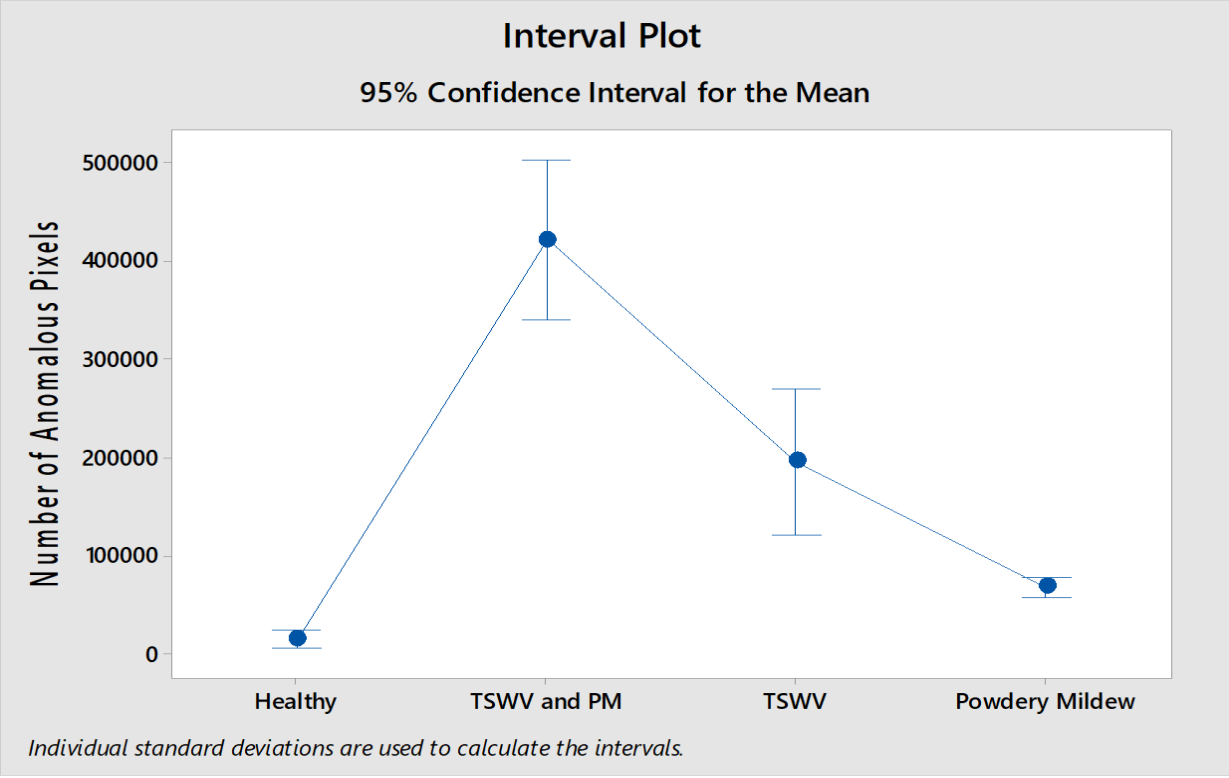


Figure 4-7 Results of the One-sided ANOVA test

Notice that, P-Value < α, hence the results are statistically significant i.e. the means are different.

Experiment images

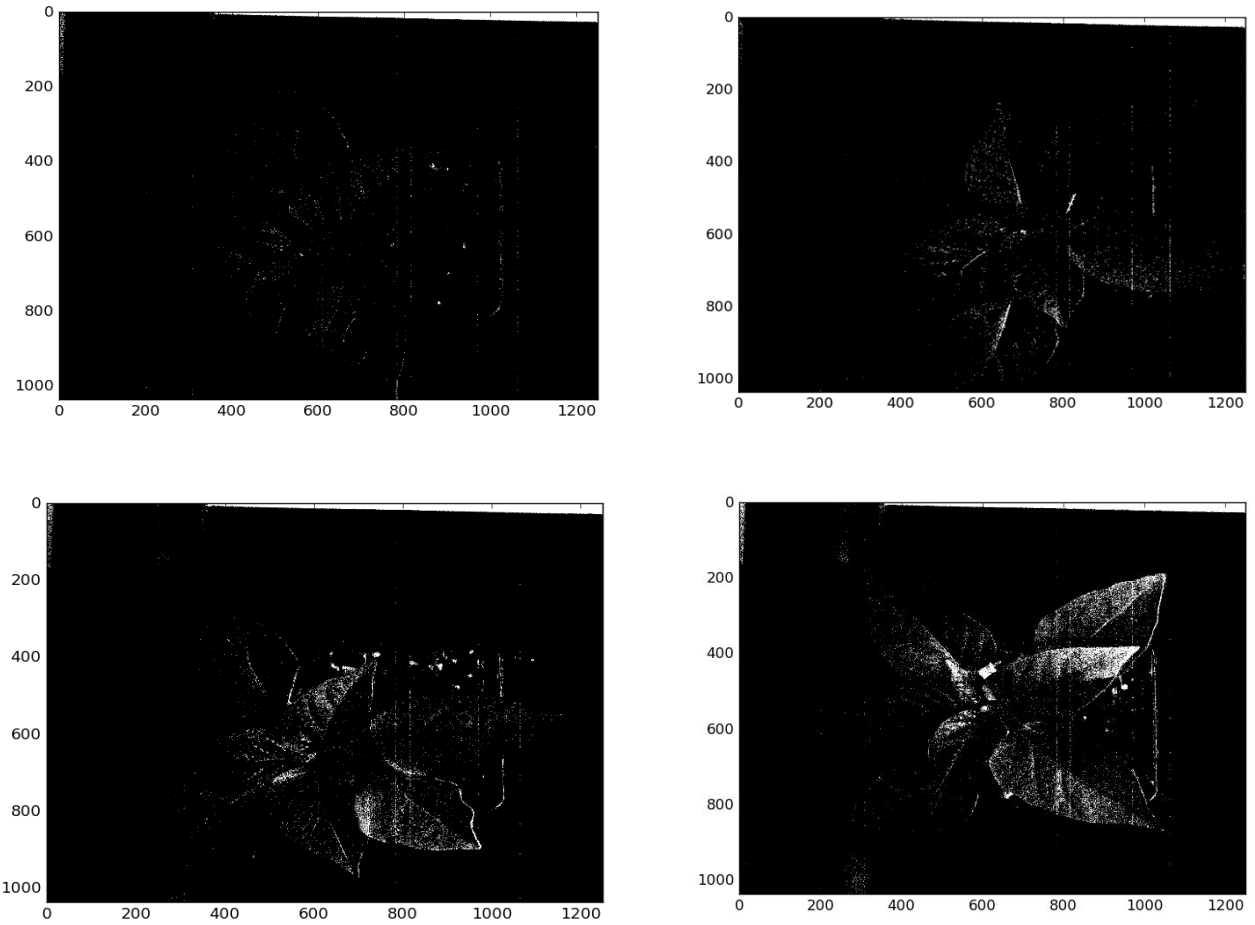


Figure 4-8 Images from the test set of Healthy plants. White indicates pixels anomalous with regard to the training set.

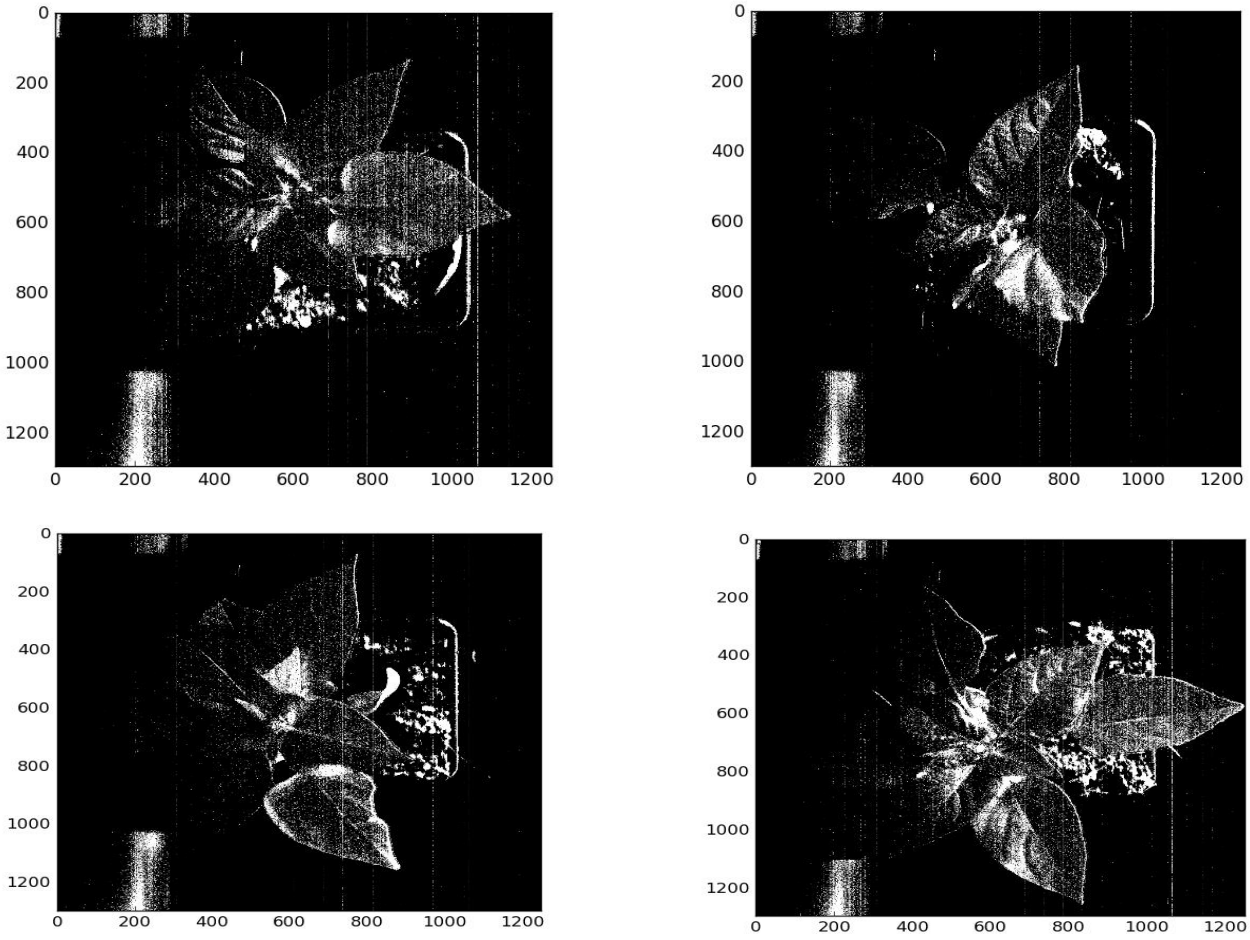


Figure 4-9 Images from the test set of plants with Powdery Mildew. White indicates pixels anomalous with regard to the training set.

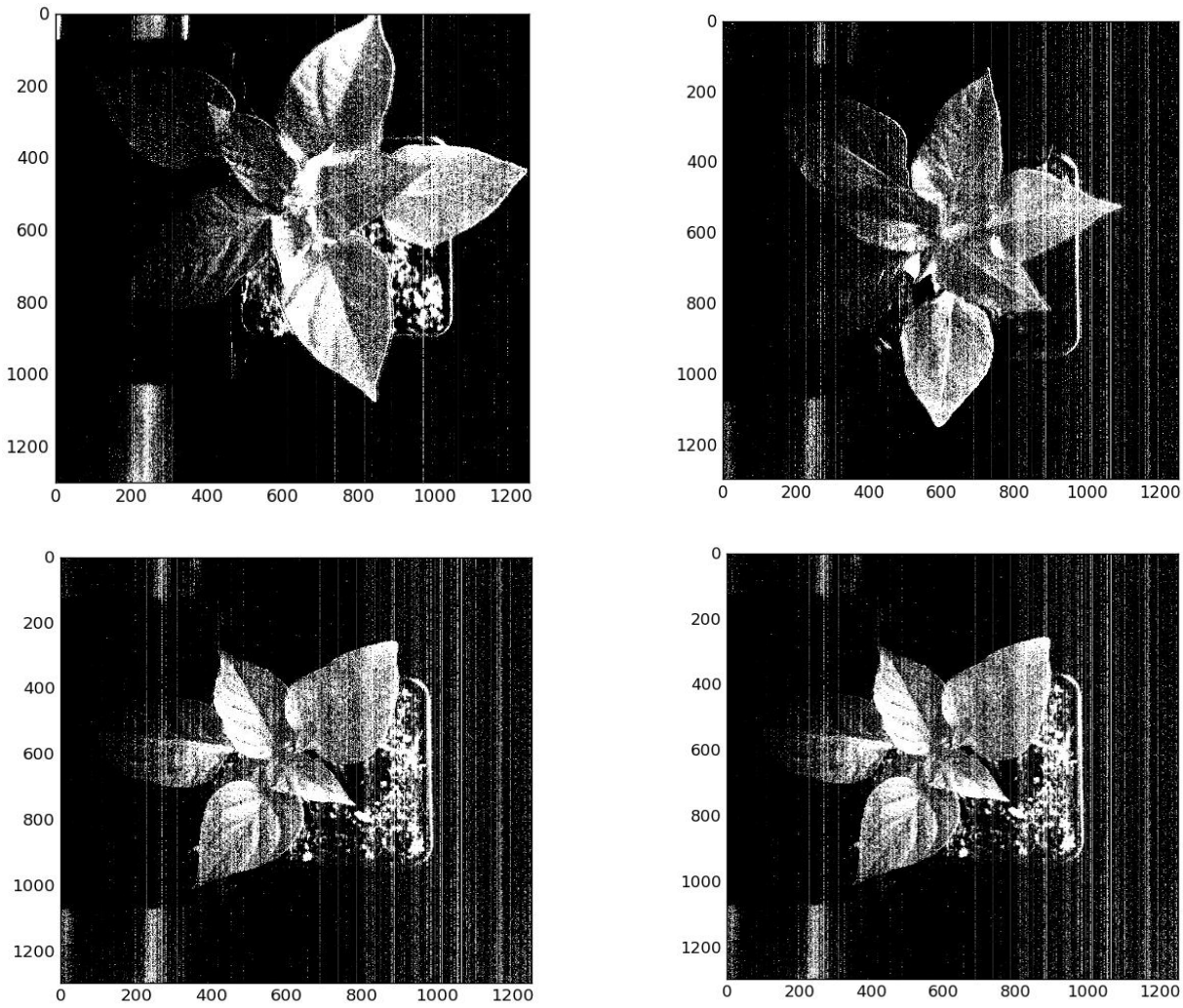


Figure 4-10 Images from the test set of plants with TSWV. White indicates pixels anomalous with regard to the training set.

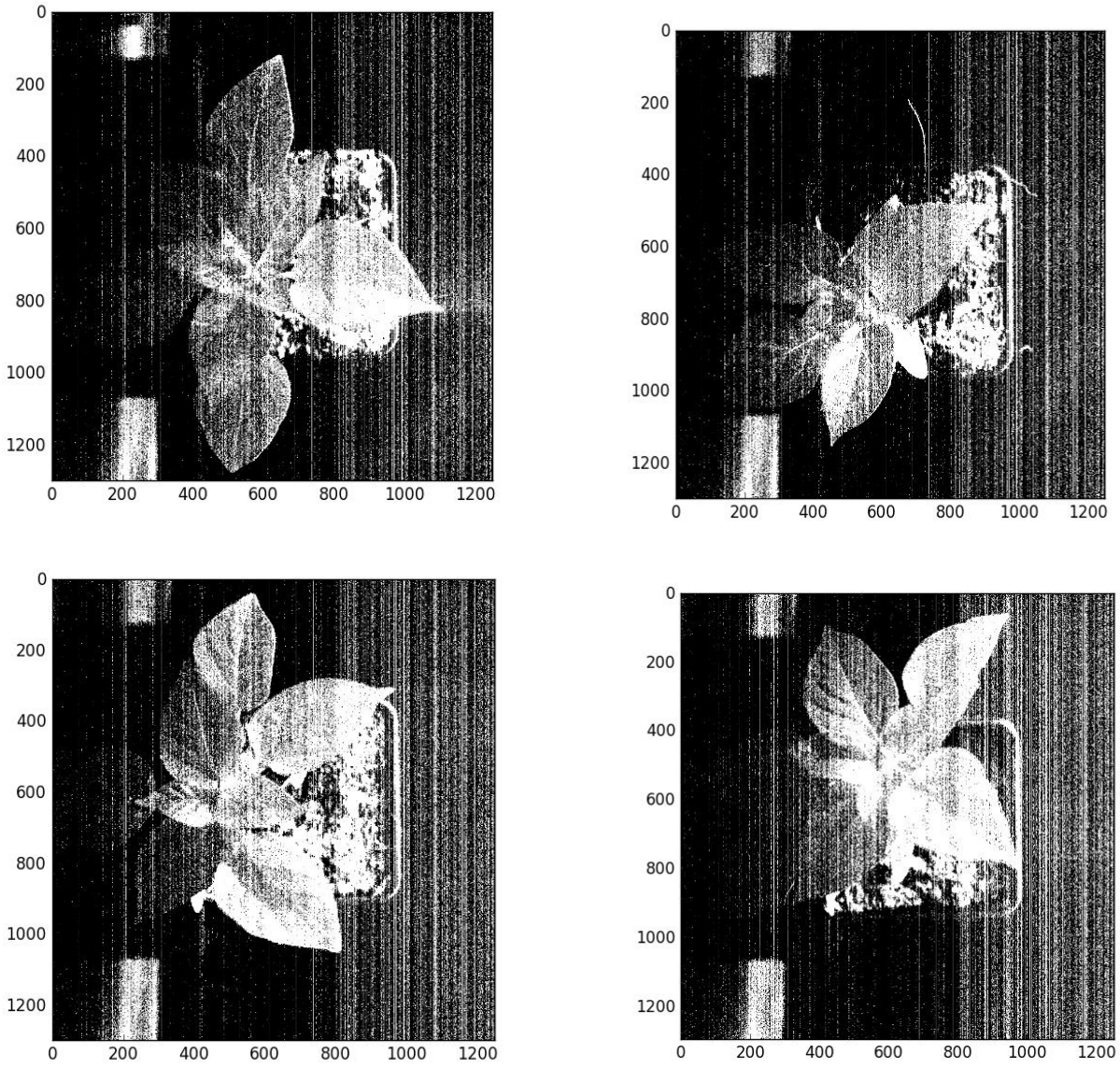


Figure 4-11 Images from the test set of plants with TSWV and Powdery Mildew. White indicates pixels anomalous with regard to the training set.

Observations

- The differences in the number of anomalous pixels detected between each of these sets were tested for statistical significance using the One-sided ANOVA test. The means are different with statistical significance at 95% confidence level. This concludes that on an

initial and basic level, the protocol can be used to identify anomalous plants in the greenhouse based on the background statistics of healthy plant images.

- The mean number of anomalous pixels in each of the categories tested progresses in the following order: Healthy < PM < TSWV < TSWV & PM. The categories are statistically significant based on tests performed on the available data.
- This result is significant as the pixels with only a probability of less than 0.001 were considered anomalous.
- Despite the random uniformly chosen spectral bands chosen to perform the analysis, the methodology to detect anomalies could differentiate between the four biotic stresses considered for this experiment. Hence it can be used to perform an initial and early anomaly detection as part of the task of monitoring the condition of pepper plants in the greenhouse.

Limitations of experiment

- The size dataset used for this implementation was 42 spectral images, 14 for training and 28 for testing which is a relatively small dataset.
- The number of anomalous pixels could also be created by the positioning of the camera over the leaf which was generally precise but there were variations
- Since the anomalies that were not based on the plant images were discounted manually, there is some uncertainty in the exact number of anomalous pixels in the image
- The exact type of anomaly or stress is expected to depend upon the spectral wavelength range used for detection. The determination of the stress and their optimal wavelengths is beyond the scope of this work.

4.4 Telerobotic control of robot in agricultural setting using HUB-CI system

- A robot cart with a Kinect camera physically located at the Agricultural Research Organization at Rishon LeZion, Israel was remotely controlled by researchers at Purdue University using the HUB-CI direct steering software (Agent A in Fig 3-2).
- Commands were sent using Python and Robotic Operating System (ROS) programs via a Google Drive. Vision was facilitated via ROS and TeamViewer software
- Average lag time of remotely sent commands was 1.06 seconds across 2 different sets of runs of 30 minutes each.

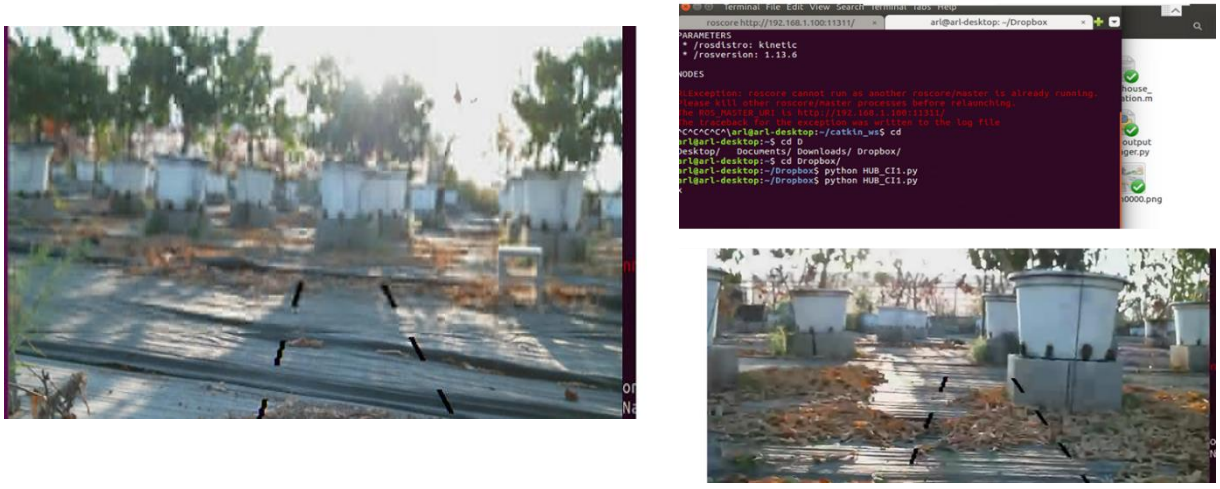


Figure 4-12 Remote Teleoperation of the robot in an agricultural setting using the HUB-CI system

5. CONCLUSION AND DISCUSSION

5.1 Advantages of Organized Collaboration via HUB-CI in an agricultural domain

The following is a summary of benefits of using a HUB based system for networked management of an agricultural greenhouse system:

1) Better overall system efficiency as demonstrated by Experiment 1. In terms of time to complete and detections detected, organized collaboration has been demonstrated as providing superior performance.
2) Capability of agents to gather inputs from multiple other agents in a distributed network or from multiple knowledgebases on the web
3) Planning of greenhouse activities can be performed collaboratively by remote and local agents with access to the same Decision Support Tools.
4) Optimized matching of tasks to agents based on availability, capacity, capability etc.
5) Based on near real time information, teams of agents to handle a particular task can be formed with agents able to Associate or Disassociate with the concerned team. Issues with the greenhouse can be resolved via the formation of dynamic teams of agents via cyber augmented methods
6) Synchronization of machine intelligence with human intelligence in real time to perform agricultural and maintenance activities in the greenhouse. Generation of optimal collaboration and task requirement schedules based on near real time inputs from sensors and human operators of different domain expertise.
7) The CI-tool of statistical anomaly detection on spectral images of plant can be applied for the early detection of biotic and abiotic stresses in plants in the greenhouse as demonstrated by Experiment 2.

5.2 Objectives and Research Questions and how they were addressed

The following table describes the objectives of this research and how they were addressed:

Table 5.1: Contribution of this research

	Objective	**Research Question**	**Research Contribution**
1)	To create a HUB based Human Robot Interaction (HRI) system that facilitates collaboration between a) Agricultural Experts, b) Human operators, c) multiple software agents related to navigation, control and disease detection, all of which are not at the same geographical location i.e. remote agents/operators.	RQ 1	The HUB-CI-DSS for Agricultural Robotic systems with all the functions described in Sections 3.2 to 3.6
2)	To develop learning-based protocols to enhance Human Robot Interaction and Decision Support capabilities of the Integrated Planner	RQ 2	The learning-based protocols to detect imaging anomalies and perform segmentation (Protocol 1), and the protocol to detect anomalous plants (Protocol 2) described in Sections 3.6, 4.1, 4.2 and 4.3.
3)	What DSS tools are necessary for a) optimal collaboration and minimal error in an agricultural setting, and b) enable early detection of stresses in plants?	RQ 3	The collaborative anomaly detection protocol to detect any anomalies based on the spectral images of leaves (Protocol 2), and the collaborative clustering protocol (Protocol 1) for imaging described in Sections 3.6, 4.1, 4.2 and 4.3

5.3 Limitations and Future Research

This research has the following limitations:

1) HUB-CI Function 2 i.e. “Workflow Optimization- task and data dependencies” has not been tested rigorously based on realistic simulations or real-world implementations.
2) HUB-CI Function 4 i.e. “Best Matching of Networked System Agents to Tasks” has been defined but not yet tested rigorously based on realistic simulations or real-world implementations.
3) HUB-CI Function 5 i.e. “Handling Conflicts and Errors” has been well defined and articulated but not yet tested rigorously based on realistic simulations or real-world implementations.
4) A real-world implementation of the system has only been performed for direct steering and remote teleoperation tasks but not yet for imaging, and early detection of stresses.
5) The collaboration strategy outlined in this work has been simulated with sensitivities to multiple factors as shown in Section 4.1 but has not yet been tested in a real-world greenhouse or agricultural setting.

Future work in this area would include the following:

1) Full scale testing of the system described between Sections 3.2 to 3.8 in a real-world agricultural environment in order to assess the value added by each of the HUB-CI functions and protocols, and the effectiveness of the collaboration strategy.
2) Rigorous simulations of HUB-CI Functions 2, 4 and 5 would be required to determine effectiveness and improve implementation.
3) Creation of “Recommender Systems” that learns the past agent decisions and activities under specific conditions and make recommendations to operators in the present. This would be a beneficial towards the overall goal of Decision Support.

APPENDIX A. CODE

A.1. To detect anomalous plants from spectral images (Python programming language)

```
import os
import time
import numpy # Import python package for scientific computing
from scipy.stats import chi2  #Import the chi square distribution function
from spectral import * # Import all python spectral imaging functions
import spectral.io.envi as envi
# Generate matrix for storing background statistics
mean_mat = numpy.zeros((10))
var_mat = numpy.zeros((10,10))
# To open several hyperspectral images in a database
train_length = 21
f11 = "15.10.16_8am_Healthy_#"
f12 = 0
f13 = "_up_HS.hdr"
f21 = "15.10.16_8am_Healthy_#"
f22 = 0
f23="_up_HS.raw"
i = 0
for i in range(1,train_length + 1):
    print(i)
    f1 = f11 + str(i) + f13
    f2 = f21 + str(i) + f23
    temp_img =  envi.open(f1,f2)
    sample = temp_img[:, :,0:840:84] # Uniform selection of bands from each image
    # Calculate background statistics i.e. mean and covariance
    A = calc_stats(sample ,mask=None, index=None, allow_nan=False)
    temp_mean = A.mean
```

```
    mean_mat = mean_mat + temp_mean
    temp_var = A.cov
    var_mat = var_mat + temp_var
# Store background statistics as variables
m = mean_mat/train_length
cv = var_mat/train_length
t_stats = GaussianStats(mean=m, cov=cv)

# Open a sample test image
img1 = envi.open('day1healthy2_2016-07-25_06-49-44_botritis.hdr','day1healthy2_2016-07-
25_06-49-44_botritis.raw')
sample1 = img1[:,:,0:840:10]

rxvals = rx(sample1, t_stats) # Calculate the Mahalanobis distance of the sampled image
                              # with respect to the background statistics
P = chi2.ppf(0.99, nbands)    # Determine the pixels with probability less than 0.01
                               # using Chi Square distribution

v = imshow(1 * (rxvals > P)) # Display the most anomalous pixels
v = imshow(rxvals)           # Display entire image based on the Anomaly Score
```

A.2. To determine if sufficient image has been taken (Python programming language)

```
import os
import time
import numpy                 # Import Python package for scientific computing
from spectral import *       # Import all python spectral imaging functions
import spectral.io.envi as envi
import gc
length = 8
# To open several hyperspectral images in a database
f11 = "15.10.16_9+6dai_8am_TSWV+PM_#"
```

```
f12 = 0
f13 = "_up_HS.hdr"
f21 = "15.10.16_9+6dai_8am_TSWV+PM_#"
f22 = 0
f23="_up_HS.raw"
i = 0
for i in range(1,length + 1):
    print(i)
    f1 = f11 + str(i) + f13
    f2 = f21 + str(i) + f23                 # Calculate k-means clustering given the
    temp_img =  envi.open(f1,f2)            # maximum number of clusters and the
    sample1 = temp_img[:,:,0:840:30] # maximum number of iterations for specified images
    (m, c) = kmeans(sample1, 12, 50)
```

A.3. Simulation for Collaboration Strategy using HUB-CI DSS (Python programming language)

```
import numpy as np
def check_defect(P,x,y):            #function to simulate
                                    # robot's search for disease or plant defect
    if (P[x][y] == 1) or (P[x][y] == 2):
        return [P[x][y], x, y]
    else:
        return []
def generator(P):
    p1=[0.8, 0.1, 0.1] # area modeled as mostly healthy plants
    p2=[0.1, 0.8, 0.1] #area modeled as mostly diseased plants
    p3=[0.1, 0.1, 0.8] #area modeled as mostly plants with unknown anomalies
    for i in range(0,40):
        if i < 14:
            p = p1
        if (i >= 14) and (i < 28):
```

```
            p = p3
        if i >= 28:
            p = p2

        r = np.random.choice(3, 40, p=p)
        #print(r)
        P[i,:]= r
    return [P,0,13,14,27,28,40]
def traverse(P): #Function for Step 3
    timer = 0
    defects = []
    for i in range(0,len(P)):
        for j in range(0,40):
            R = check_defect(P,i,j)
            if len(R) ==0 :          # check if Robot or HO has returned a
                                     # defect
                pass
            elif len(R) == 3:
                defects.append(R)
                timer = timer + 1
    #print(len(defects))
    return [timer, len(defects)]
def sampler(P): #Function for Step 4
    rows = len(P)
    cols = len(P[0])
    sample_size = int(cols*rows*0.1)
    random_x = np.random.randint(0,rows, size=sample_size)
    random_y = np.random.randint(0,cols, size=sample_size)
    timer = 0
    for i in range(0,sample_size):
```

```
            check_defect(P,random_x[i],random_y[i])
            timer = timer + 1
    return timer

def sampler2(P): # Function for Step 5
    rows = len(P)
    cols = len(P[0])
    sample_size = int(cols*rows*0.45)
    random_x = np.random.randint(0,rows, size=sample_size)
    random_y = np.random.randint(0,cols, size=sample_size)
    timer = 0
    timer_robot = 0
    timer_human = 0
    defects = []
    for i in range(0,sample_size):
            if timer > int(sample_size/2):
                timer_human = timer_human + 1
            else:
                timer_robot = timer_robot + 1

            R= check_defect(P,random_x[i],random_y[i])
            defects.append(R)
            timer = timer + 1
    return [timer_robot, timer_human, len(defects)]
def main():
    P = np.random.randint(0,3, size=(40, 40)) #Generate plants as diseased,
                                    # anomalous, or healthy randomly
    [P,hl,hul,al,aul,dl,dul] = generator(P)
     P = P.tolist()
     P_2 = P[hl:hul]
```

```
    robot_P = P[dl:dul] # plants with high disease probabilities
    human_P = P[al:aul] # plants with "unknown anomalies/other defects"
    [timer_robot_1, defects_R1] = traverse(robot_P)
    [timer_human_1, defects_H1] = traverse(human_P)
    timer_human_2 = sampler(robot_P)
    [timer_robot_2, timer_human_3, defects_RH2] = sampler2(P_2)

   print(defects_R1 + defects_H1 + defects_RH2)
   print(timer_robot_1 + 2*timer_human_1 + timer_human_2 + timer_robot_2 +
2*timer_human_3)
```

A.4. Simulation for Baseline Scenario – No HUB-CI DSS (Python programming language)

```
import numpy as np
def check_defect(P,x,y):          #function to simulate
                                  # robot's search for disease or anomaly
    if (P[x,y] == 1) or (P[x,y] == 2):
        return [P[x,y], x, y]
    else:
        return []
def check_defect_HO(P,x,y):          #function to simulate
                                  # human's search for disease or anomaly
    if (P[x,y] == 1) or (P[x,y] == 2):
        return [P[x,y], x, y]
    else:
        return []
def increment(P,x,y, dflag):  # Function for simulating robot cart
                          #traversing greenhouse
    cflag = True
    x_max = P.shape[0] - 1
    y_max = P.shape[1] - 1
    if dflag is True:
```

```
        x_new = x+1
        y_new = y
        cflag = False
        if x_new > x_max:
            x_new = x
            y_new = y+1
            dflag = False
            cflag = False
    if cflag is True:
        if dflag is False:
            x_new = x - 1
            y_new = y
            if x_new < 0:
                x_new = x
                y_new = y+1
                dflag = True
                cflag = False
    if y_new > y_max:
        x_new = False
        y_new = False
        dflag = False
    return [x_new,y_new,dflag]


def main():
    P = np.random.randint(0,3, size=(40, 40)) #Generate plants as diseased,
                                    # anomalous, or healthy randomly
#    P = generator(P)
    x = 0
    y = 0
    dflag = True
```

```
tflag = True
defects = []
timer_robot = 0
timer_human = 0
while (x or y) or tflag:              #Run the program and above functions
    [x,y,dflag] = increment(P,x,y,dflag)
    R = check_defect(P,x,y)
    if len(R) ==0 :                   # check if Robot or HO has returned a
        #print(x,y, R)                # defect
        pass
    elif len(R) == 3:
        defects.append(R)
    tflag = False
    timer_robot = timer_robot + 1
for defect in defects:
    typ = defect[0]
    x = defect[1]
    y = defect[2]
    timer_human = timer_human + 1
print(len(defects))
print(timer_robot + 2*timer_human + 10)
```

REFERENCES

1) Sudduth, K. A. (1998). Engineering for Precision Agriculture-Past accomplishments and future directions (No. 982040). SAE Technical Paper.

2) An, W., Wu, D., Ci, S., Luo, H., Adamchuk, V., & Xu, Z. (2017). Agriculture Cyber-Physical Systems. In Cyber-Physical Systems (pp. 399-417).

3) Edan, Y., Han, S., & Kondo, N. (2009). Automation in agriculture. In Springer handbook of automation (pp. 1095-1128). Springer, Berlin, Heidelberg.

4) Khaitan, S. K., & McCalley, J. D. (2015). Design techniques and applications of cyberphysical systems: A survey. IEEE Systems Journal, 9(2), 350-365.

5) "US National Science Foundation, Cyber-Physical Systems (CPS)

6) McLennan, M., & Kennell, R. (2010). HUBzero: a platform for dissemination and collaboration in computational science and engineering. Computing in Science & Engineering, 12(2), 48-53.

7) Nof, S. Y., Ceroni, J., Jeong, W., & Moghaddam, M. (2015). Revolutionizing Collaboration through e-Work, e-Business, and e-Service (Vol. 2). Springer.

8) Nof, S. Y. (Ed.). (2009). Springer handbook of automation. Springer Science & Business Media.

9) Bechar, A., & Edan, Y. (2003). Human-robot collaboration for improved target recognition of agricultural robots. Industrial Robot: An International Journal, 30(5), 432-436.

10) Zhong, H. (2012). HUB-based Telerobotics (Doctoral dissertation, Purdue University).

11) Zhong, H., Wachs, J. P., & Nof, S. Y. (2014). Telerobot-enabled HUB-CI model for collaborative lifecycle management of design and prototyping. Computers in Industry, 65(4), 550-562.

12) McLennan, M., & Kennell, R. (2010). HUBzero: a platform for dissemination and collaboration in computational science and engineering. Computing in Science & Engineering, 12(2), 48-53.

13) McLennan, M., Clark, S., Deelman, E., Rynge, M., Vahi, K., McKenna, F., ... & Song, C. (2015). HUBzero and Pegasus: integrating scientific workflows into science gateways. Concurrency and Computation: Practice and Experience, 27(2), 328-343.

14) Wang, Z., Gong, L., Chen, Q., Li, Y., Liu, C., & Huang, Y. (2016, August). Rapid Developing the Simulation and Control Systems for a Multifunctional Autonomous Agricultural Robot with ROS. In International Conference on Intelligent Robotics and Applications (pp. 26-39). Springer, Cham.

15) Emmi, L., Paredes-Madrid, L., Ribeiro, A., Pajares, G., & Gonzalez-de-Santos, P. (2013). Fleets of robots for precision agriculture: a simulation environment. Industrial Robot: An International Journal, 40(1), 41-58.

16) Zhong, H., Levalle, R. R., Moghaddam, M., & Nof, S. Y. (2015). Collaborative intelligence-definition and measured impacts on internetworked e-work. Management and Production Engineering Review, 6(1), 67-78.

17) Devadasan, P., Zhong, H., & Nof, S. Y. (2013). Collaborative intelligence in knowledge-based service planning. Expert Systems with Applications, 40(17), 6778-6787.

18) Song, D., Goldberg, K., & Chong, N. Y. (2008). Networked telerobots. In Springer Handbook of Robotics (pp. 759-771). Springer, Berlin, Heidelberg.

19) Bechar, A., Meyer, J., & Edan, Y. (2009). An objective function to evaluate performance of human–robot collaboration in target recognition tasks. IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews), 39(*6*), 611-620.

20) Rodriguez, G., & Weisbin, C. R. (2003). A new method to evaluate human-robot system performance. Autonomous Robots, 14(2-3), 165-178.

21) Moghadam, P., Ward, D., Goan, E., Jayawardena, S., Sikka, P., & Hernandez, E. (2017, November). Plant Disease Detection Using Hyperspectral Imaging. In Digital Image Computing: Techniques and Applications (DICTA), 2017 International Conference on (pp. 1-8). IEEE.

22) Mahlein, A. K., Rumpf, T., Welke, P., Dehne, H. W., Plümer, L., Steiner, U., & Oerke, E. C. (2013). Development of spectral indices for detecting and identifying plant diseases. Remote Sensing of Environment, 128, 21-30.

23) AlSuwaidi, A., Grieve, B., & Yin, H. (2018). Combining spectral and texture features in hyperspectral image analysis for plant monitoring. Measurement Science and Technology, 29(10), 104001.

24) Nansen, C., Geremias, L. D., Xue, Y., Huang, F., & Parra, J. R. (2013). Agricultural case studies of classification accuracy, spectral resolution, and model over-fitting. Applied Spectroscopy, 67(11), 1332-1338.

25) Cheng, H., Peng, H., & Liu, S. (2013, March). An improved K-means clustering algorithm in agricultural image segmentation. In PIAGENG 2013: Image Processing and Photonics for Agricultural Engineering (Vol. 8761, p. 87610G). International Society for Optics and Photonics.

26) Guo, P., Dusadeeringsikul, P., & Nof, S. Y. (2018). Agricultural cyber physical system collaboration for greenhouse stress management. Computers and Electronics in Agriculture, 150, 439-454.

27) Mehta, P., Shah, H., Kori, V., Vikani, V., Shukla, S., & Shenoy, M. (2015, March). Survey of unsupervised machine learning algorithms on precision agricultural data. In Innovations in Information, Embedded and Communication Systems (ICIIECS), 2015 International Conference on (pp. 1-8). IEEE.

28) Wang, J., Miao, Y., Khamis, A., Karray, F., & Liang, J. (2016, July). Adaptation Approaches in Unsupervised Learning: A Survey of the State-of-the-Art and Future Directions. In International Conference Image Analysis and Recognition (pp. 3-11). Springer, Cham.

29) Wang, D., Vinson, R., Holmes, M., Seibel, G., Bechar, A., Nof, S., ... & Tao, Y. (2018). Early Tomato Spotted Wilt Virus Detection using Hyperspectral Imaging Technique and Outlier Removal Auxiliary Classifier Generative Adversarial Nets (OR-AC-GAN). In 2018 ASABE Annual International Meeting (p. 1). American Society of Agricultural and Biological Engineers.

30) Cheein, F. A., Herrera, D., Gimenez, J., Carelli, R., Torres-Torriti, M., Rosell-Polo, J. R., ... & Arnó, J. (2015, March). Human-robot interaction in precision agriculture: Sharing the workspace with service units. In Industrial Technology (ICIT), 2015 IEEE International Conference on (pp. 289-295). IEEE.