Testbed Report – PRISM ARS draft 2.1.18

Contents

Testbed Report – PRISM ARS draft 2.1.18

Introduction

The purpose of a testbed for this task is to study and evaluate the implementation of precision agriculture techniques that enables performance and collaboration of a team of farmers and robots to be analyzed. Execution of specific tasks can be simulated in a three-dimensional environment. An important aspect of constructing a robotic unit capable of executing agricultural tasks is the ability to integrate the necessary agricultural knowledge (Emmi, 2014). Hence development of a simulation environment that enables integration of robotics with agriculture. There are at-least two research directions in recent years that have attempted to solve this problem. They will be discussed in this report.

The robot testbeds for precision agriculture that have been reviewed so far are all pertaining to *fully autonomous* agricultural systems. However, with regard to the BARD project, the testbed must be created to test effectiveness for human-robot collaboration in precision agriculture. Agricultural applications of robotics require advanced technologies to deal with complex and highly variable environments and produce (Nof,'09) and not all horticultural applications can be fully automated in the near term. As known via previous research performed, robots are better suited for repetitive tasks compared to humans, whereas humans have better capabilities when it comes to perception, cognition and creativity. Robots will be used to report legitimate

agricultural abnormal situations and conditions as soon as they emerge. The stresses that this project seeks to consistently detect and eliminate include CGMMV, TSMV and Powdery Mildew in tomatoes, cucumbers and peppers. Agricultural human experts can receive those abnormal reports, interpret them and dispatch a robot to arrive at given locations, take pictures, and transmit the pictures back to agricultural experts. This testbed will be unique in that it will integrate the principles of Collaborative Control Theory developed by Dr. Nof and PRISM center researchers: algorithmic collaboration requirement planning, e.g., of tasks, motion and orientation trajectories, generates coherent control plans for the collaborative execution by subsystems (Nof & Chen,'03, Rajan & Nof,'96, Rajan & Nof,'96) so they will achieve: (1) simplified input/output interface with HO supervisory control (Bechar et al.,'14); (2) efficient utilization of onboard computing resources with reusable solutions and learning functions (Zhong et al.,'15), e.g., of vision-led approach motions; (3) integrated planning considering constraints from different subsystems and HO, e.g., overcoming unexpected obstacles to sensing approach (Kaelbling & Lozano-Pérez,'11); (4) collaborative error detection and prevention, or resolution, for fault tolerance, e.g., for adjusting integrated motion and sensing plans (Jeong & Nof,'09, Oren et al.,'12). Integrating HO into a robotic system will not impair 'timely detection' since most time spent in current manual inspection is on moving between locations, not due to actual manual inspection time.

Benchmarking is another aspect that the testbed can be useful for. The testbed can come in handy to compare and contrast various methodologies and algorithms for the various tasks that this project requires. With a testbed, the impact of a change in a micro aspect/task of the system on the macro agricultural system can be determined and measured. Reproducibility of experiments and conditions is another problem that can be solved by using a testbed (Yan, Laval,

Fabresse, Bouraqadi, 2017). Experiments must be reproducible in order to allow researchers to compare their approach with existing ones, where the reproducibility implies that there are no hidden assumptions so that people can reproduce the experiments using a different simulator. Moreover, the simulation should be as realistic as possible, because benchmarks are defined to evaluate applications prepared for the real world rather than a simplified version. (Yan, Laval, Fabresse, Bouraqadi, 2017).

Related Work

Fleets of robots for precision agriculture: a simulation environment (Emmi, Madrid, Ribeiro, 2013)

System description:

In this research a computational environment named "Simulation Environment for Precision Agriculture Tasks using Robot Fleets" (SEARFS) has been developed 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** and is being developed with the purpose of providing a general programming system for the following:

- Observing and evaluating different fleet of robots' configurations while simulating the execution of various agricultural tasks (e.g. a heterogeneous fleet, composed of robots equipped with chemical and mechanical systems for weed management).
- Implementing different types of sensor and actuator systems in a fleet of robots and evaluating the robot cooperation behavior.
- Generating missions for the fleet of robots, in order to observe and evaluate the acquisition of field information and the actuation for managing the field variability.
- Representing and modeling the field characteristics in a 3D virtual universe, to attain improved understanding (e.g. modeling spatial distribution of field, soil, crop and weed variabilities).

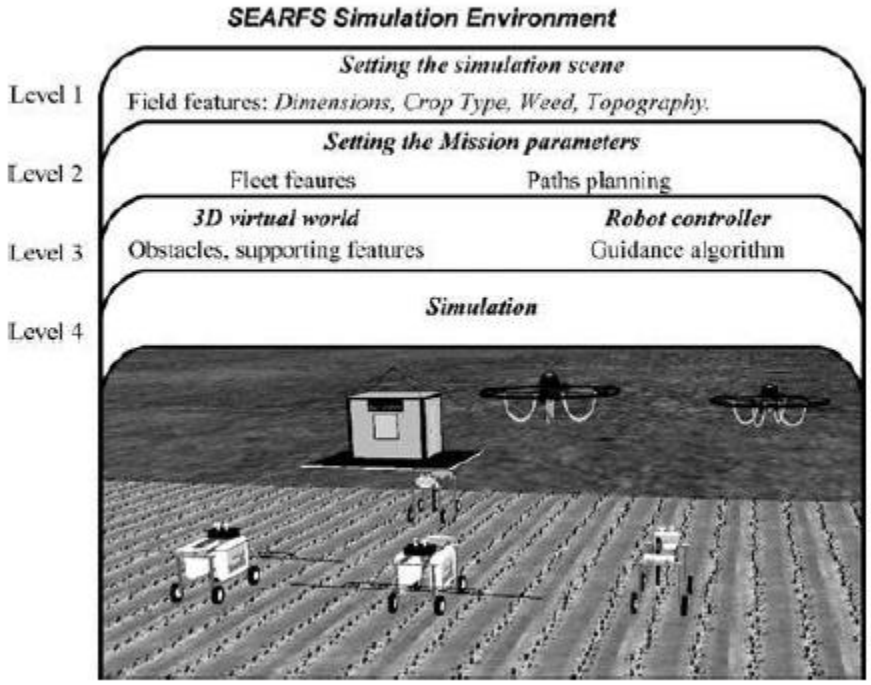

Fig 1. SEARFS environment configuration levels (Courtesy Emmi et al. 2013)

The SEARFS environment consists of four levels of configurations (above figure), where the lower levels depend on the configuration of the higher levels:

- Level 1: At this level, the user can define the characteristics of the field where the simulation is performed.
- Level 2: At this level, the user can establish the mission to be executed and the autonomous fleet of robots in the SEARFS environment.
- Level 3: At this level, the user may perform modifications on the 3D virtual world, which are previously generated based on the information of the first two levels.
- Level 4: At the final level of configuration, the user may generate simulations in which the execution of the planned mission can be represented and evaluated.

System structure:

SEARFS consists of two main modules:

- Graphical user interface (GUI): The GUI oversees making a 3D representation of the working environment, including vehicles, sensors, actuators, manipulators, crops, weeds, uneven terrain, etc. Moreover, the GUI allows the user to model, program, and simulate very complex mobile robots and manipulators that can be equipped with a variety of sensors and actuators, all interacting in a 3D virtual world with its elements. Webrobots package was used to develop the GUI interface.
- Configuration user interface (CUI): The CUI oversees defining structures and carrying out computations to allow the user to configure the working environment in an easy and simple way, as well as to enable the current information of the models representing the elements in the working environment to be expanded. MATLAB was used to develop the

CUI model.

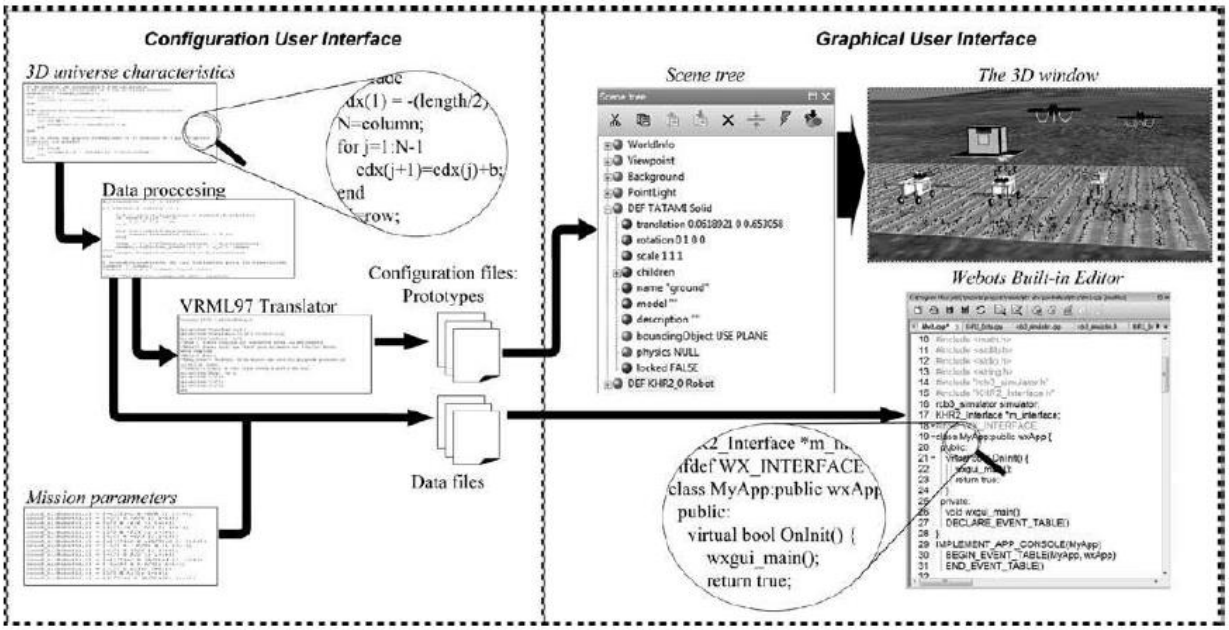

Fig 2. The two modules of SEARFS (Courtesy of Emmi et al. 2013)

Availability of SEARFS software package for the BARD project

Users can download the latest version of SEARFS from the developer's web site (www.car.upm-csic.es/searfs). The web site also contains all the necessary links to execute the application (MATLAB, Webots, etc). To define the elements of the virtual world and to configure the fleet mission, the user must run the CUI executable file first. The user should follow a set of steps in order to define the characteristics of the virtual field correctly, as well as the fleet configuration and the trajectories of each mobile unit.

Conclusions and value for BARD project:

The system presented above attempts to unify two different perspectives – robotics and agriculture, with the objective of studying and evaluating realistically the implementation of Precision Agriculture techniques in a 3D virtual world.

Several robot simulation tools used currently (Emmi et al. 2013)

Table 1

Simulator	**Description**
Gazebo	3D multi-robot simulator with dynamics, capable of simulating articulated robots in complex and realistic environments
Simbad	Java 3D robot simulator for studying situated AI, machine learning (AI algorithms), in the context of autonomous robotics and autonomous agents
Microsoft Robotics Developer Studio	Visual simulation environment that enables users to develop robots in a rich virtual environment with realistic physics and state-of-the-art rendering
Robot studio, ROS	Open-source, meta-operating system for robots. Provides services such as hardware abstraction, low-level device control, implementation of commonly used functionality, message-passing between processes, and package management
Webots	Development environment used to model, program and simulate mobile robots. The user

	can design complex robotic setups, with one or several similar or different robots, in a shared environment

And most importantly here are some simulation tools that are used in Precision Agriculture applications (Emmi, 2013):

Simulator	Description
Agriculture Production Systems Simulator (APSIM)	APSIM is a tool for exploring agronomic adaptations such as changes in planting dates, cultivar types, and fertilizer/irrigation management
Cropping Systems Simulation Model (CropSys)	CropSys is a system that is used as an analytical tool to study the effects of climate, soil, and management in agricultural production systems and the environment
The Decision Support System for Agrotechnology Transfer (DSSAT)	A software package that combines the effects of soil, crop phenotype, weather and management options which allow the user to get results by conducting simulation experiments
Simple and Universal Crop Growth Simulator (SUCROS)	SUCROS is a mechanistic model that explains crop growth on the basis of the

	underlying processes. It simulates potential growth of a crop and can also describe production under water-limited conditions by including water balances of crop and soil

HUB-CI based collaborative testbed for Precision agriculture

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, Goldberg, and Chong 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?

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 standard web browser and utilize high performance Grid computing resources (Zhong, Nof 2013). HUB along with cloud computing allows software and data to be easily shared by groups of users. The majority of HUBs allow collaborative jobs on virtual materials and simulations, but there is no tool for users to perform physical collaboration (Zhong, Nof 2013). HUB-CI addresses the problem of managing physical collaboration between groups of human users plus cyber physical agents. HUB-CI, i.e., HUB-based Collaborative Intelligence is a set of collaborative intelligence algorithms and tools that have been developed to enhance HUB and augment productivity with more efficient functions to support collaborations (Zhong, Nof 2013).

How has it been applied previously, and how can it be applied to the BARD testbed?

It has been applied in the following research:

A collaborative telerobotics network framework with hand gesture interface and conflict prevention (Zhong, Nof, Wachs 2013)

This research performed by (Zhong, Nof, Wachs 2013) provides:

- Collaboration framework for the remote operation of a telerobotics system with a collaborative control algorithm aggregating multiple users' inputs according users' performance, and
- A decentralized conflict/error prevention network to assure safe and effective teleoperation.
- A new vision-based hand gesture recognition algorithm for human–robot interface

This work has provided a distributed framework for collaborative control in telerobotic systems. The deployed collaborative telerobotic system has six major components, i.e., the operators, the web user interface, the **CEDP (Conflict and Error Detection and Prediction/Prevention)** agent at the user side, the collaboration support (CS) agent, the CEDP agent at the robot side, and the robots (Zhong, Nof, Wachs 2013). Human operators give commands to the robots and robot arm via hand gestures. The workflow for this process is as follows (Zhong, Nof, Wachs 2013):

Step 1: One operator launches the interface and evokes a gesture to command the telerobots.

Step 2: A video frame is captured by the user interface

Step 3: Every frame is processed through a computer vision sub-system for gesture recognition

Step 4: The recognized gesture command is diagnosed by the Conflict and Error Detection and Prevention (CEPD) agent at the user side to prevent errors. If the command passes the CEDP agent's diagnosis, the command is sent to the server.

Step 5: After all operator's commands are transmitted to the server, the commands are integrated through an aggregation scheme by the CS agent and delivered to robots.

Step 6: The CEDP agent at the robot side analyses the aggregated commands according to the robot's current status.

Step 7: If the aggregated commands are error free, they are uploaded to robots and executed.

The diagram of this process is given in the figure below (Zhong, Nof, Wachs 2013):

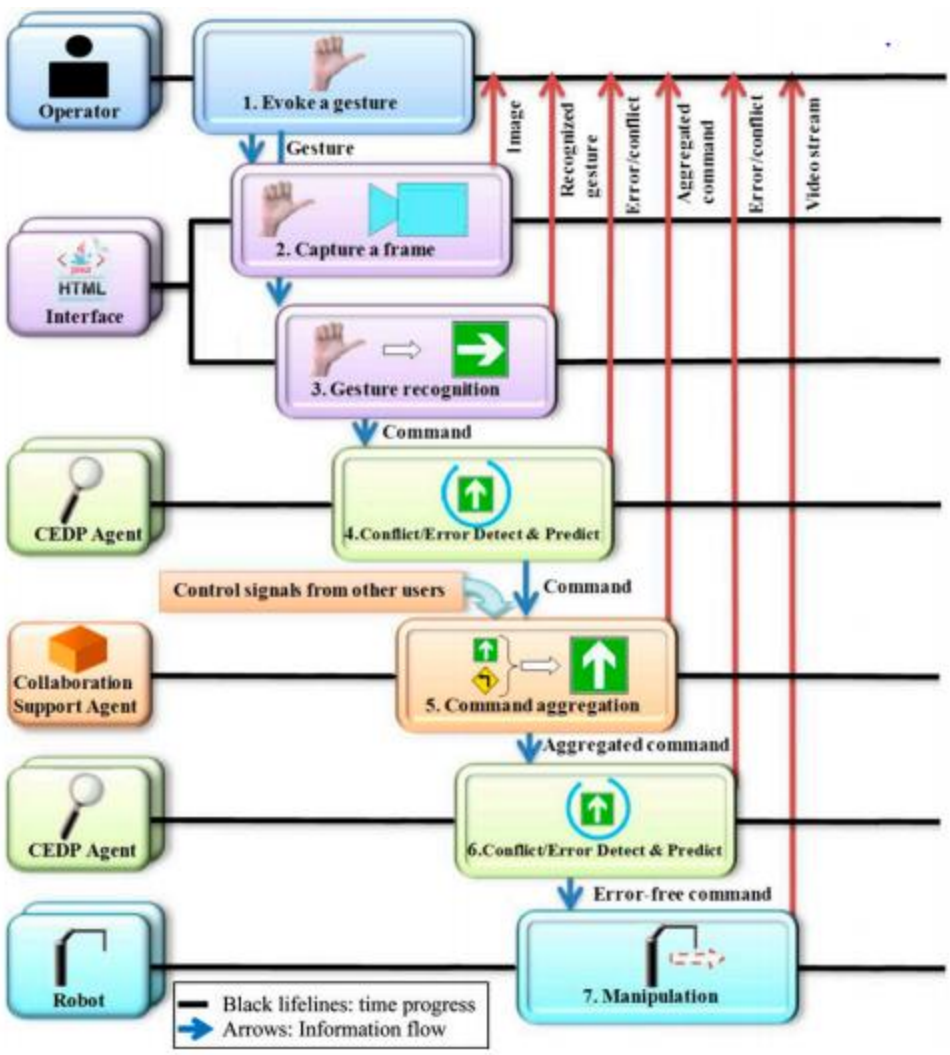

Fig. 3 Collaborative Telerobotics system sequence diagram (Courtesy of Zhong et al. 2013)

Conflict and Error Prevention (CEPD) is crucial to the telerobotic system being developed for the BARD project as the overall performance of the telerobotic system is affected by conflicts/errors. In this research (Zhong, Nof, Wachs 2013) CEPD agents were implemented in the system. A CEDP network is designed according to a general CEDP model (Chen and Nof 2010) to prevent conflicts/errors as early as possible. **An error** occurs when collaborators' behavior does not meet their objectives or capabilities. **A conflict** is such behavior that does not

meet their peers' objectives. The following diagram describes the structure of conflict/error detection and prediction network (Zhong, Nof, Wachs 2013):

Fig.4 Structure of conflict/error detection and prediction network

Command aggregation for collaborative control: This aspect of the research would be applicable to BARD project where there are **multiple Human Operators with varying levels of expertise**.

In unstable and noisy network connections, multiple operators also provide backups and redundant signals for each other. By aggregating gesture commands from multiple operators through the collaborative support agent, the CTR (Collaborative Tele-Robotic) system can minimize the risk of disconnections or conflicts generated by errors. Collaborative control is thus fault tolerant and can yield better, more effective results from a team of weak agents relative to a single, even faultless agent (Nof 2007). In this research (Zhong, Nof, Wachs 2013), the collaborative control logics for collaborative tasks are determined according to specified applications (Ko and Nof 2012). The proposed aggregation logic is a weighted voting algorithm of discrete control signals. The aggregated command to be assigned to robots is decided based on the weighted votes from all operators.

This is the element of HUB-CI that has been applied in this research and can be highly useful for the BARD project.

Experiments and results: Individual and collaborative experiments have been designed to evaluate the performance of the CTR system. A virtual robot arm model has been developed in 3Ds Max to mimic the manipulator used for a nuclear decommissioning task. The robotic task consists of controlling two robots in order to perform a pipe disassembling task in the decommissioning of a nuclear power station. A total of 11 operators participated in the experiment. Four of them were part of the cohort used for the hand gesture recognition training dataset. The participants were novice operators. During collaboration, all operators were seated in the same room with different computers and could not see or talk to each other after the experiment started. Collaborators performed gestures simultaneously and controlled the same set of robots together on the same task. The performance measurements of the proposed system include the time to completion, conflicts and errors, average interval between error-free commands, manipulation distance, and total commands. It was found that the commands are more accurate for the robotic task in the collaborative experiments compared to those in the novice operators' individual experiments where the CEDP agents mitigate errors and conflicts. The increased accuracy in commanding robots is due to the collaboration support agent which aggregates the control signals from two operators (Zhong, Nof, Wachs 2013). When an expert made an error, but the novice operator issued an error-free command, the novice operator's weight was larger than the expert's weight. Due to this collaboration scheme, the system can be tolerant to a control signal with continuous errors.

Agricultural Cyber Physical System framework for greenhouses by Dr Guo, Oak P. Dusadeerunsikul and Dr. Nof and its testbed implementation

Dr Guo, Oak P. Dusadeerunsikul and Dr. Nof 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, aimed at monitoring, detecting, and responding to abnormal situations and conditions with the purpose of providing 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. The following figure describes the Agricultural CPS framework (Guo, Dusadeerunsikul and Nof 2017):

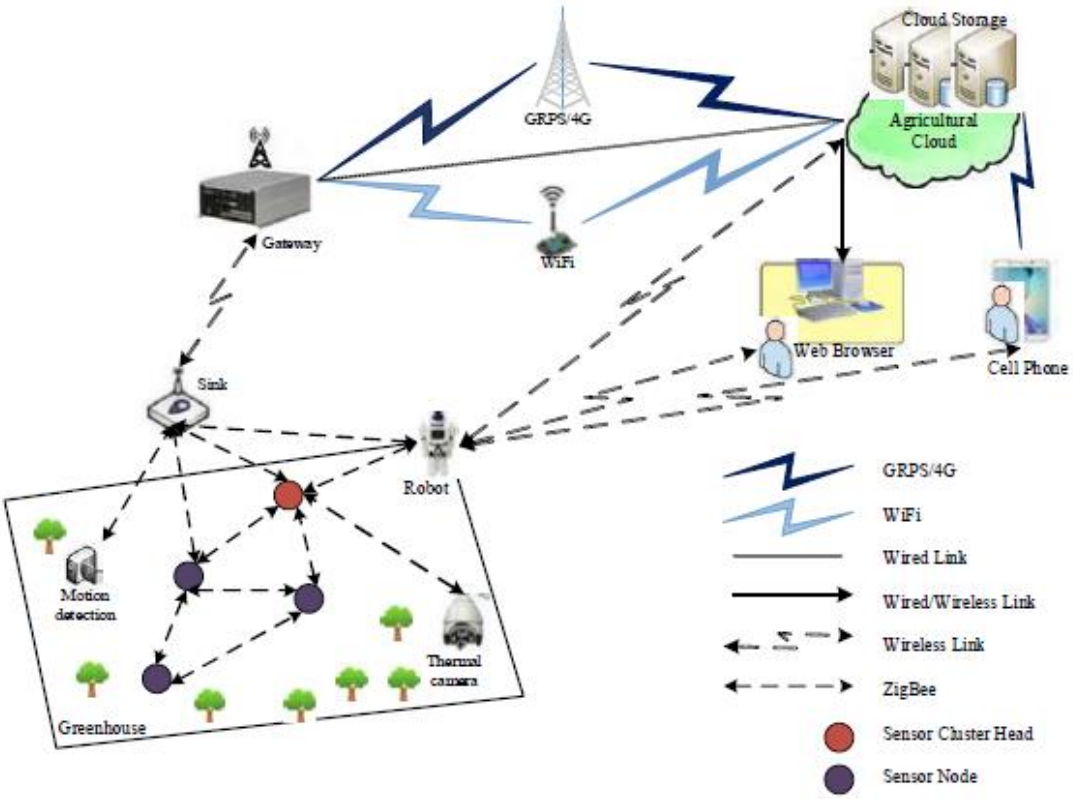

Fig 5. Agricultural CPS Framework (Courtesy of Guo, Dusadeerunsikul, Nof, 2017)

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,

Dusadeerunsikul and Nof 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, Dusadeerunsikul and Nof 2017):

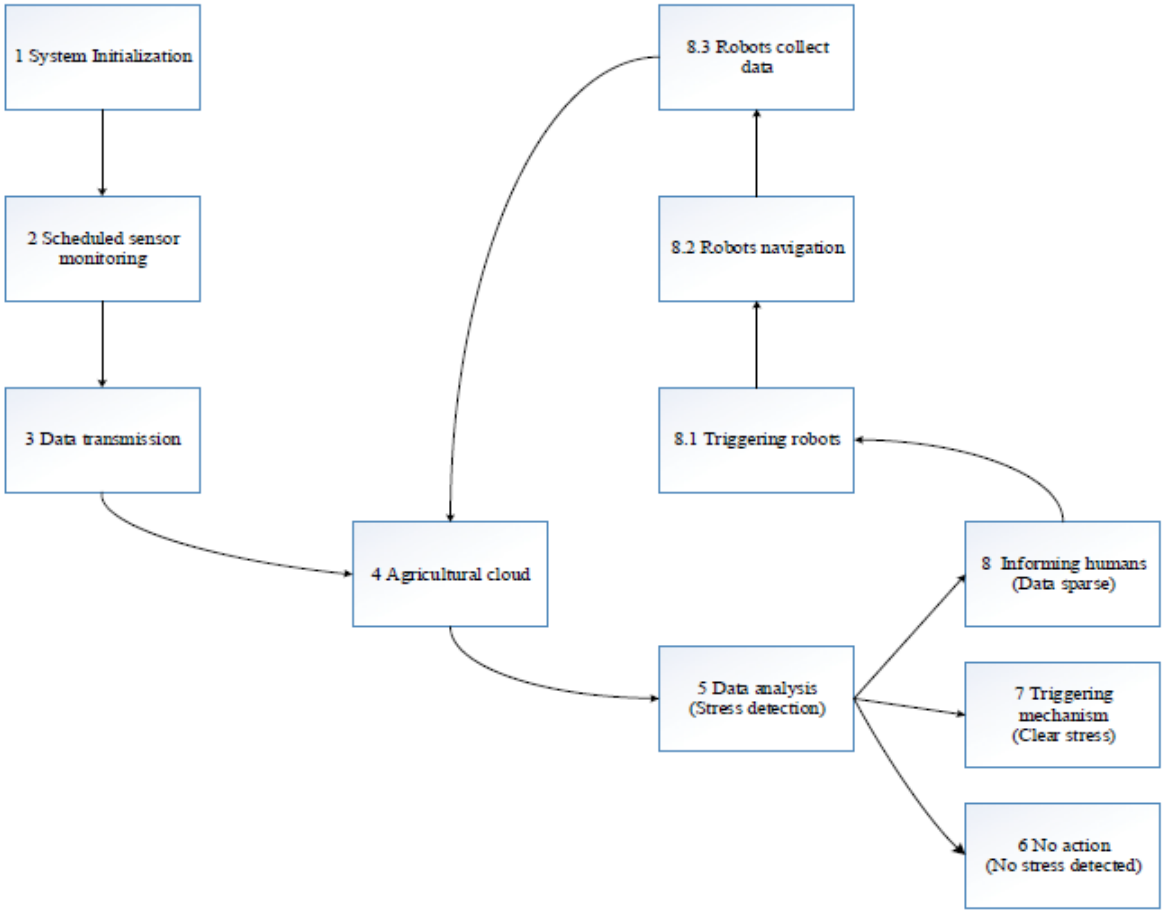

Fig. 6. MDR-CPS workflow diagram (Courtesy of Guo, Dusadeerunsikul, and Nof, 2017)

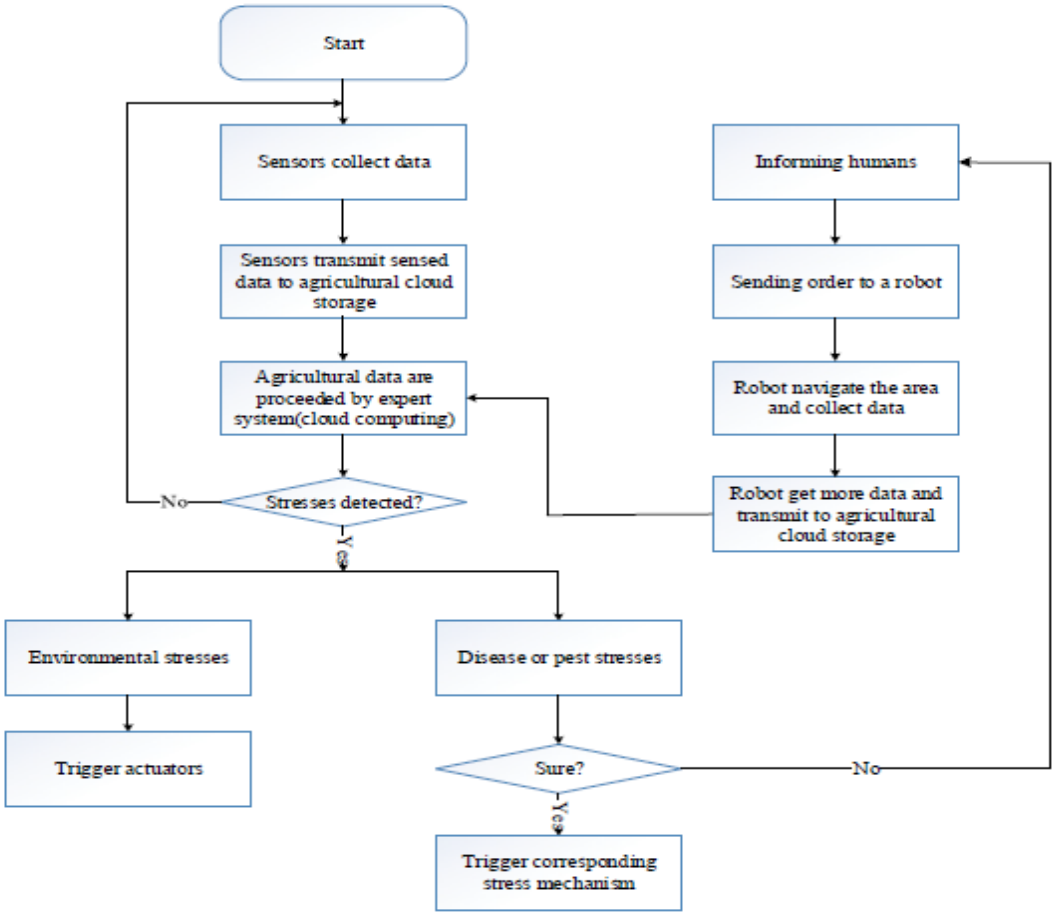

Fig. 7. MDR-CPS workflow chart (Courtesy of Guo, Dusadeerunsikul, and Nof, 2017)

The above MDR-CPS architecture can be implemented on a ROS/Gazebo platform or the SEARFS simulation testbed platform described earlier in this report.

PRISM/ARO current Testbed Design

As mentioned in earlier reports, a testbed can be developed via Robotic Operating System (ROS) and Gazebo robot simulator. Manual control of the robot can be done via several ROS programs/packages. The Routing, conflict and error prevention algorithm(s) (developed by PRISM/ARS Oak Puwadol D. (2018) and Praditya Ajidarma (2017) as part of this project) can be programmed as a ROS package via Python or C++. The following diagram describes the various elements of the testbed and the relationships between each of them. Note that the various elements of the system interact via the ROS middleware.

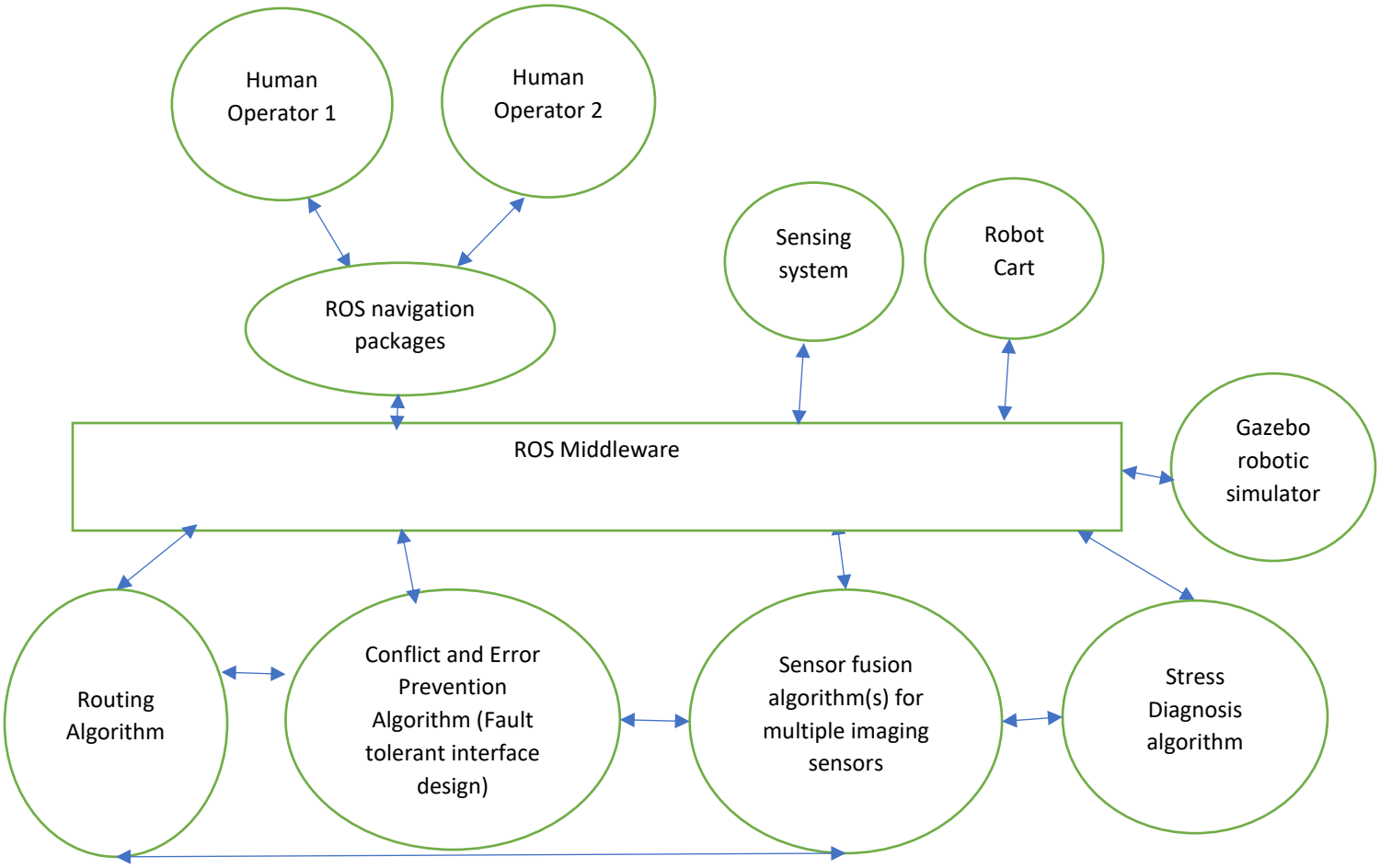

Fig 8. Elements of the testbed

Two types of testbeds can be created:

1) Physical testbed with ROS middleware: This testbed would include the robot cart and the sensor system and a network of computers (developed by Dr. Tao' UMD team and Dr. Bechar's ARL team) all connected on the same network and coordinated via a ROS architecture.
2) Simulated testbed with ROS middleware: This testbed would include a robotic simulator (preferably Gazebo robotic simulator) on which the robot cart, sensor system and a simulated physical environment can be created. The Gazebo software has a good physics engine and the environment can be created accurately using this ROS feature.

1) Physical testbed with ROS middleware

Table 2 Mapping the proposed testbed (1) to our BARD Project tasks

	Components	**Status**	**Corresponding BARD Research Project task(s)**
1)	Robot cart	Available	All tasks
2)	Time of Flight camera, stereo camera, RGB camera etc.	Available	3.5.1.b. Sensor fusion study and optimal sampling
3)	Real greenhouse environment (with tomato, cucumber and pepper plants)	Available	All tasks

4)	One or more human operators	Available	3.5.1.4 Human-Robot collaboration system
5)	Automated routing algorithms	Can be completed in 1 – 2 weeks	3.5.1.4.b Development of adaptive algorithms
6)	Conflict and error prevention algorithms	Can be completed in 1 – 2 weeks	3.5.1.4.d Fault-tolerant interface design
7)	ROS navigation packages	Completed	3.5.1.4.b Development of adaptive algorithms
8)	ROS mapping packages	Almost complete	3.5.1.4.b Development of adaptive algorithms
9)	ROS human-robot interaction interface	Can be completed in 1 week	3.5.1.4 Human-Robot collaboration system
10)	ROS human operator interaction interface	Mode undecided (Suggestions offered in this section)	3.5.1.4 f. Collaboration requirements algorithms/protocols

11)	Database capability	Preliminary version available in ARO	All tasks
12)	Sensor fusion and machine learning algorithms/packages to determine location in mapping	Can be completed in 2 weeks	3.5.1.4.b Development of adaptive algorithms
13)	Machine learning algorithms to gain insights on HO inputs and automated system outputs	Not completed yet (Not an immediate requirement)	3.5.1.2c Decision support algorithms 3.5.1.4 Human-Robot collaboration system
14)	Sensor fusion for plant images taken by sensors	Required from ARO and UMD	3.5.1.2c Decision support algorithms
15)	Algorithm to diagnose stress	Required from ARO or UMD	3.5.1.2c Decision support algorithms
16)	Single integrated network on which all systems will run	Not ready	3.5.1.4 Human-Robot collaboration system

For the time being *human operator interaction amongst each* can be performed via video conferencing software (Skype, Webex) till a collaborative online HUB software tool can be purchased or developed.

<u>Simulated testbed with ROS middleware</u>

The robot can be controlled on the Gazebo simulator and in real using the keyboard, and also can be controlled by automated programs written in Python or C++. ROS was used to perform this task. So as of now, we can implement the new algorithms for automatic routing with adaptive search and we can also manually control the robot via keyboard.

Robotic mapping of the environment can also be performed. We are able to remotely use the mapping packages on ROS to have the robot create maps of the greenhouse. Furthermore, we can also visualize a remote robot's environment using rviz. The next step in this task will be to give the robot command goals, i.e., command the robot to go to a particular location specified by the user (farmer) on the map.

Here are the various elements of the testbed:

Table 3 Mapping Testbed components and our BARD project tasks

	Components of testbed	**Status**	**Corresponding BARD Research Task**
1)	Simulation software functioning with ROS middleware (Gazebo software)	Completed	3.5.1.5b Development of a simulation tool
2)	Virtual box robot cart for simulation (created using xml)	Completion < 2 days	3.5.1.5b Development of a simulation tool

3)	Virtual sensors (time of flight camera, stereo camera, RGB camera etc.) (created using xml)	Time of flight completed	3.5.1.2b Sensor fusion study and optimal sampling
4)	Simulated greenhouse environment (Virtual plants and greenhouse layouts)	Completed	3.5.1.5b Development of a simulation tool
5)	One or more human operator(s)	Available	3.5.1.4 Human-Robot collaboration system
6)	Automated routing algorithms	Can be completed in 1 – 2 weeks	3.5.1.4.b Development of adaptive algorithms
7)	Conflict and error prevention algorithms	Can be completed in 1 – 2 weeks	3.5.1.4.d Fault-tolerant interface design
8)	ROS navigation packages	Completed	3.5.1.4.b Development of adaptive algorithms
9)	ROS mapping packages	Almost complete	3.5.1.4.b Development of adaptive algorithms

10)	ROS human-robot interaction interface	Can be completed in 1 week	3.5.1.4 Human-Robot collaboration system
11)	ROS human operator interaction interface	Mode undecided (Suggestions offered in this section)	3.5.1.4 f. Collaboration requirements algorithms/protocols
12)	Database capability	Preliminary version available in ARO	All tasks
13)	Sensor fusion algorithms/packages to determine location in mapping	Can be completed in 2 weeks	3.5.1.4.b Development of adaptive algorithms
14)	Machine learning algorithms to gain insights on HO inputs and automated system outputs	Not completed yet (Not an immediate requirement)	3.5.1.2c Decision support algorithms 3.5.1.4 Human-Robot collaboration system
15)	Sensor fusion for plant images taken by sensors	Required from ARO	3.5.1.2c Decision support algorithms
16)	Algorithm to diagnose stress	Required from ARO or UMD	3.5.1.2c Decision support algorithms

17)	Single network on which all systems will run	Not ready	3.5.1.4 Human-Robot collaboration system
18)	Gazebo web interface for collaborative simulations	Can be completed in 1 week	3.5.1.5b Development of a simulation tool

Note: Several tasks outlined above can be performed concurrently, and completion times are conservative estimates.

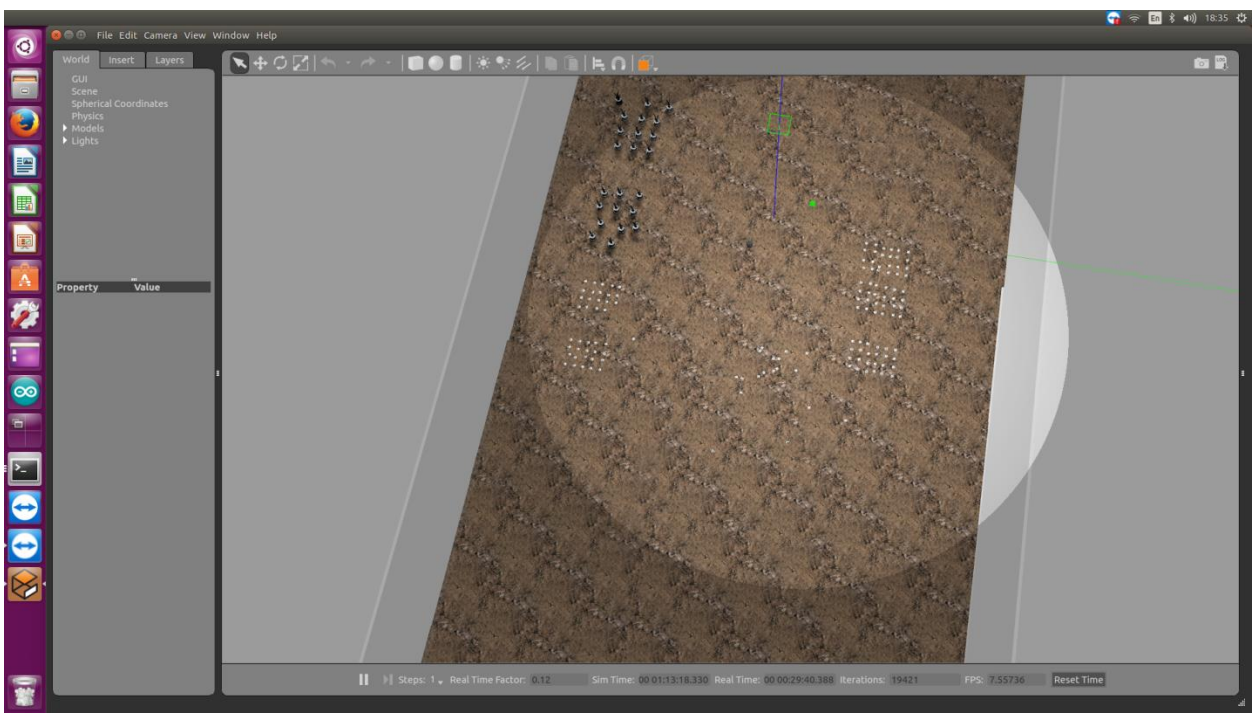

Fig.9. Top view of the greenhouse simulation

ROS topics and nodes record all data involving the simulation including sensor data. ROS messages can be saved for analysis.

Challenges expected in creating the testbed

1) Sensor fusion algorithm for camera images and stress diagnosis algorithm: In order to perform meaningful experiments with the images in either the ROS simulated, or the ROS based real world testbeds, these two algorithms that lead to diagnosis of the stress detected.
2) Mapping of the environment: For the human operator(s) to provide navigation goals to the robot in both the physical and simulated testbed, a map of the environment must be available. This map can be created by navigating the robot through the physical or virtual area while the robot sensors gather information of the surroundings which includes obstacles. Then sensor fusion can be used to synchronize the sensor information with those of the odometry readings and hence create an accurate map. Using the map, the human operator may command the robot to move to specific locations based on whatever their requirements are.
3) Network challenges: If the testbed is going to be a physical and not simulated one, then in order for the ROS based testbed to work, all the participating computers must be running on the same network. This would be challenging as the participating computers are in multiple geographical locations. The only way to connect them would be by creating a Virtual Private Network. This was unsuccessfully attempted in September 2017 for connecting the PRISM machine with the robot computer at ARO in Israel. In order to succeed at this task a paid VPN service would need to be set up. However, if we are using the simulated test bed via the Gazebo web interface, then network is not an issue anymore.

Collaborative simulations

Collaborative simulations can be performed on the Gazebo simulator. Using the gazebo web feature Gzweb, users on different machines can interact with the simulation. Gzweb is a WebGL client for Gazebo. It's a front-end graphical interface to gzserver and provides visualization of the simulation. However, Gzweb is a thin client in comparison, and lets you interact with the simulation from the comfort of a web browser. This means cross-platform support, minimal client-side installation, and support for mobile devices.

As an alternative, TeamViewer can also be used for the 3 research centers to interact with the simulations simultaneously.

Testing plan for the Human-Robot Collaboration (HUB-CI) in the BARD project

Possible performance metrics to measure the effectiveness of the system include but are not limited to the following:

1. Overall time to complete detection and identification tasks (TTC) i.e. Time taken for an overall run.
2. Time taken by operator to confirm or alter system's decision (T_h)
3. Number of Conflicts and Errors detected, as a function of time detected.
4. Probability that sensor n will be utilized ($p_{s(n)}$)
5. Probability that the automated system will cause error (p_{ms})
6. Probability that the human operator will cause error (p_{mh})
7. Expected number of disagreements in diagnosis between human and automated system (E)

8. Actual number of disagreements between human operator and automated system in run i (A_i)
9. Operational costs of the system (C_o)
10. The average time interval between error free commands during a task (MTBE, Mean Time between Error free commands).

Suggested Possible Experiments

Measure **TTC, Operational cost, and Average Time interval between two error free commands** of the ROS based HUB system for the following scenarios:

1) Test the integration of the Fault tolerant interface and Adaptive routing algorithm on a random scenario run in ROS for the above performance metrics. (BARD research tasks: 3.5.1.4 b) and d))
2) Test the integrated Fault tolerant Interface, the Adaptive Routing Algorithm, the Collaboration requirements/algorithms and protocols, and sensor fusion results while providing random inputs for disease identification by the system and that by more than one operators, again for the above performance metrics. (BARD research tasks: 3.5.1.4 b), d) and f), and 3.5.1.2b))
3) Test the integrated Fault Prevention Interface, the Adaptive Routing Algorithm, the Collaboration requirements/algorithms and with several images combined from the sensors via sensor fusion where operators make decisions on which sensors must be used, the orientation of the sensors/cameras, and also for decisions regarding disease identification. (BARD research tasks: 3.5.1.4 b), d) and f), and 3.5.1.2b))

With each planned experiment, the level of complexity of the system increases.

Research hypotheses and testbed capabilities

In the research plan, three hypotheses were made. The following summarizes the hypotheses and compares it with the capabilities of the testbed proposed in this report.

Hypothesis 1

This hypothesis states that multi-modality imaging sensor fusion of thermal, 3D and multispectral imaging will improve the following three areas with minimal human supervision:

1) Detection of common biotic stresses such as Powdery Mildew, CGMMV, TSWV in plants
2) Detection of nutrient level, e.g., nitrogen, magnesium or potassium in plants.
3) Detection of water stress, that appear in greenhouse crops like cucumber, pepper and tomato.

This is a UMD and ARO task that can be tested using the proposed physical testbed HUB-CI based with ROS middleware, assuming the hyperspectral cameras or cameras with the required spectral bands are available. The flow of data can be done via the ROS based middleware.

This hypothesis cannot be tested on the proposed simulated testbed as there are no simulation softwares that can simulate plant details to the levels that are required for this project tasks. Neither the SEARFS nor Gazebo robotic simulation softwares can accurately model the spectral properties of biotic stresses, nutrient or water levels.

Hypothesis 2

This hypothesis states that the human-robot (HRI) collaborative systems will outperform both fully autonomous and manual systems in agricultural environments, when robots with sensors, fast computing, and moderate machine intelligence replace just tedious labor-intensive walk and repeated inspection motions, while humans supplement intelligence and knowledge that a robot may request/require. The HRI system identification and locating performance will be:

1) More effective and less costly than a fully robotic system (e.g., high hit rates, low false alarm rates)
2) Its total performance including detection and operation time will be faster than a human worker, or a fully robotic system

This is a PRISM center task that can be tested using the proposed physical testbed HUB-CI based with ROS middleware. A realistic experiment, howver, will depend highly on realistic performance indicators and error/conflict probability distributions provided by UMD and ARL.

This task can also be tested by simulation using some of the software tools described here and also by using simulated data.

Hypothesis 3

This hypothesis states that integrated task planning for the entire system (sensors, human, and robot) will yield better performance in handling complex early identification and locating tasks on a timely basis, with several layers of subtasks.

This is a joint task for all three research centers, and can be tested via the physical HUB-CI based collaborative system with the ROS middleware, and simulations can also be prepared and implemented.

References

1) Zhong, H., Wachs, J. P., & Nof, S. Y. (2013). HUB-CI model for collaborative telerobotics in manufacturing. IFAC Proceedings Volumes, 46(7), 63-68.
2) Zhong, H., Wachs, J. P., & Nof, S. Y. (2013). A collaborative telerobotics network framework with hand gesture interface and conflict prevention. International Journal of Production Research, 51(15), 4443-4463.
3) Yan, Z., Fabresse, L., Laval, J., & Bouraqadi, N. (2017). Building a ROS-Based Testbed for Realistic Multi-Robot Simulation: Taking the Exploration as an Example. robotics, 6(3), 21.
4) 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.
5) Matena, V., Bures, T., Gerostathopoulos, I., & Hnetynka, P. (2016, May). Model problem and testbed for experiments with adaptation in smart cyber-physical systems. In Software Engineering for Adaptive and Self-Managing Systems (SEAMS), 2016 IEEE/ACM 11th International Symposium on (pp. 82-88). IEEE.
6) Guo, P., Dusadeerunsikul, O.P., & Nof, S. (2017). Agricultural CPS Collaboration for Greenhouse Stress Management, submitted to *Computers and Electronics in Agriculture*.
7) Zhong, H. (2012). Hub-based telerobotics (Master's thesis). Purdue University.
8) Ajidarma, Praditya (2017). *Multi-Sensor Fault Tolerant Learning Algorithm in an Agricultural Robotic System*, MS Thesis, Purdue University, School of Industrial Engineering.

9) Oak Puwadol Dusadeerunsikul, Shimon Y. Nof, and Avital Bechar (2018). Detecting Stresses in Crops Early by Collaborative Robot-Sensors-Human System Automation. Article submitted to *IISE Annual Conference*, Orlando, May, 2018.