A REMOTE INTERFACE FOR A HUMAN-ROBOT COOPERATIVE VINEYARD SPRAYER

R. Berenstein, I. Ben Halevi, Y. Edan

Department of Industrial Engineering and Management
Ben-Gurion University of the Negev
Beer-Sheva, Israel

ABSTRACT

In this paper we present a remote interface for a human-robot cooperative vineyard sprayer. The interface allows the human operator to control the robot from a remote location for several tasks including vehicle navigation, target marking, target spraying and overall supervision on the task completion.

The full characterization of the interface is presented including human factors (e.g., user awareness, GUI interface), robot control parameters (e.g., robot speed\orientation, operation state), and human-robot collaboration parameters (e.g., level of autonomy to the robot, human supervision).

The interface software included two parts, the remote graphical interface and the robot management software. These two parts are connected to a single NAT and communicate using TCP\UDP protocols. The remote graphical interface was designed using WIMP (Windows, Icons, Menus, and Pointer) interface and was implemented in Java. The robot management software was developed in C++.

Keywords: HRI, robotics, user interface design, selective sprayer

INTRODUCTION

Pesticides are an integral part of the worldwide agriculture. Between 30 and 35% of crop losses can be prevented when harmful insects and diseases are eliminated by spraying pesticides (Cho and Ki, 1999). By spraying the pesticides selectively toward the targets with a robot sprayer up to 60% of pesticide use can be reduced (Elkabetz et al., 1998; Gil et al., 2007; Goudy et al., 2001). Selective sprayers provide also labour reduction, a bottleneck in many agricultural operations. Since pesticides are poisonous (Betarbet et al., 2000; Dasgupta et al., 2007; Eddleston et al., 2002; Rogan and Chen, 2005), the removal of the human from the spraying process will contribute to the farmer health.

For economic feasibility the robotic sprayer must be able to detect and spray more than 95% of the targets successfully (Blackmore et al., 2001).

Despite intensive R&D in detection of agricultural objects, most target detection applications (grapes, apples, tomatoes, oranges, peaches, melons, eggplants, and strawberries) result in only 75–80% detection of targets with a maximum of 90% successful detection noted (Berenstein et al., 2010; Jeon et al., 2005; Lamm et al., 2002). In a spraying application it is important to minimize also the false alarms so as to reduce overall pesticide material and minimize environmental pollution. In order to overcome this 95% target detection barrier, a human operator was added to the target detection process formerly carried out by the robotic sprayer using target detection algorithms (Berenstein et al., 2010). The human-robot collaboration assuming that improved system performance can be achieved by taking advantage of human perception capabilities and the robot accuracy and consistency.

The primary goal of this work is to create a user interface design to remotely control agricultural spraying robot. This work is part of a larger project aiming to develop an agricultural spraying robot.

INTERFACE CHARACTERIZATION

The interface characterization was based on the requirements extracted from the vineyard robotic sprayer platform. The system requirements are classified into two classes of importance (Ben Halevi, 2011): **high importance class**, requirements that are crucial to the overall system functionality, **low importance class**, requirements that should be met to a certain extent on the prototype and for the final commercial product and therefore these requirements are desirable but not crucial to the system functionality. Table 1 and Table 2 summarize the system UI for the high importance class and for the low important class respectively.

Table 1 – High importance class requirements

Requirement	**Description**
UI (User Interface)	ergonomic ui is required in order to create intuitive user experience while using the interface
WIMP (Windows, Icons, Menus and Pointers) based interface	the ui should be based on wimp interface since this method is the most well-known interface paradigm among worldwide users (Van Dam, 1997)
user is “the boss”	the system should allow the user to override algorithm decisions
feedback	interface should include real-time feedback regarding the overall system state (fuel, battery power, pesticide amount left)
human navigation along the vineyard rows	since the robot is designed to travel along the vineyard rows, the human should have the option to navigate the robot from a remote location using the interface
user awareness	the interface should be designed in a way for the human to have high overall awareness of the robot state
reliability	the system reliability should be extremely high since the robot is intended to work remotely from the human

	operator
emergency stop button	emergency stop button must be included in interface
obstacle detection and alert	the robot will stop when sensing that there is an obstacle and alert the user. the scope of this work does not include obstacle detection but the infrastructure for combining one should exist in the interface

Table 2 – Low importance class requirements

Requirement	**Description**
reports	the system produces operational reports
history database	the system will keep history data both for system restore and post work inquiry
help	help menu should be offered to the user in any case of operational uncertainty
sound	the system should include sound alerts
web	the system should be communicate over the web in addition to single NAT (as currently working)
backup	periodically backups will be performed in order to protect important data in case of main computer crash
parallel robots	the user should have the option to control in parallel several robots
user manual	the system should include a well-documented user manual

USER INTERFACE DESIGN

Fig. 1 depicts a crude division of the system into three high-level blocks. Each block is described separately in the following.

Robot side

The robot's operation can be roughly divided into two parts, manual control and automatic control. According to Berenstein et al. (2010), the system analyzes raw images from the camera and marks the grape clusters that should be sprayed. To increase the target detection rate the human operator interferes if necessary through the interface and helps the robot with the target detection process with three optional human-robot collaboration methods (Berenstein and Edan, 2012). This is the only non-automatic stage in the cycle. Afterwards, the system calculates the real grape clusters coordinates to be sprayed and updates the database and interface. This cycle continues until the robot reaches the end of the row. Fig. 2 presents the operational pipeline of the robot.

The Robot side software was developed using C++ language. The choice of using C++ was mainly for future software conversion to ROS (Robotic Operating System) which is C++ based.

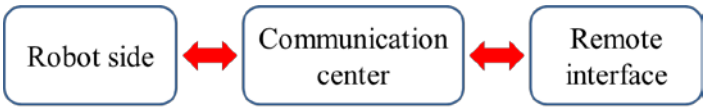


Fig. 1. High level design

Remote interface

In order to create a modular system, the interface was implemented as a layer model according to the Model-View-Controller (MVC) software architecture (Deacon, 2009). This pattern isolates "domain logic" (the application logic for the user) from the user interface (input and presentation), permitting independent development, testing and maintenance of each module.

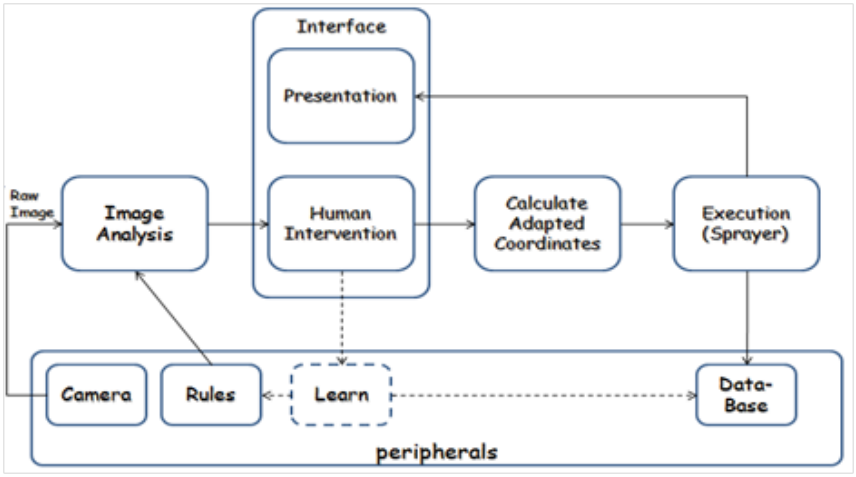


Fig. 2. Robot side pipeline

The **model** manages the behavior and the data of the application domain, responds to requests for information about its state (usually from view), and responds to instructions to change state (usually from the controller). In event-driven systems, the model notifies observers when the information changes so that they can react. The **view** renders the model into a form suitable for interaction, typically a user interface element. Multiple views can exist for a single model for different purposes. A viewport typically has a one to one correspondence with a display surface and knows how to render to it. The **controller** receives user input and initiates a response by making calls on model objects. A controller accepts input from the user and instructs the model and viewport to perform actions based on that input.

As depicted in Fig. 3, the interface side includes three logical layers. Each layer communicates only with its adjacent layers. The Presentation layer is a collection of several views that are aware of a controller which in turn mediates the views requests to the robot.

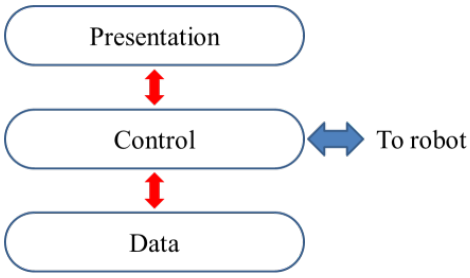


Fig. 3. Layer model

Communication center

The communication model implemented is a unique hybrid of client-server and P2P architectures conforming to the user's requirements and needs. The suggested model uses the robot as a server, waiting for the interface as a client at port 6789. After establishing connection and configuration of the robot, both robot and interface begin to act as 2 peers in a P2P network, exchanging messages between them. The P2P architecture is implemented as double parallel client-server architectures.

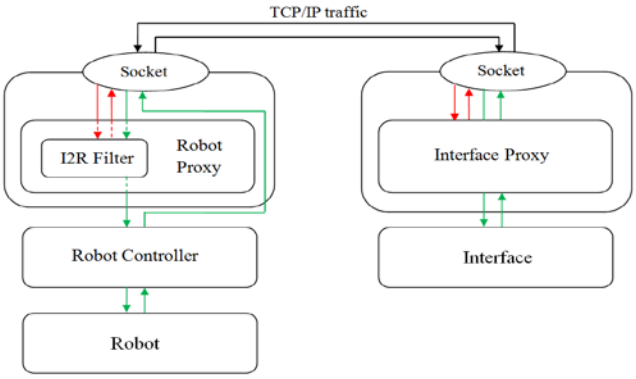


Fig. 4. Communication model

USER INTERFACE IMPLEMENTATION

The user interface is a set of windows with different functionally designs to help the user control the robot and to help the robot with the target detection task.

Security window. The security window is the first window that appears to the user when the system is activated. The security window includes only two fields for the user to fill, the “User Name” and the “Password” field (Fig. 5.a).

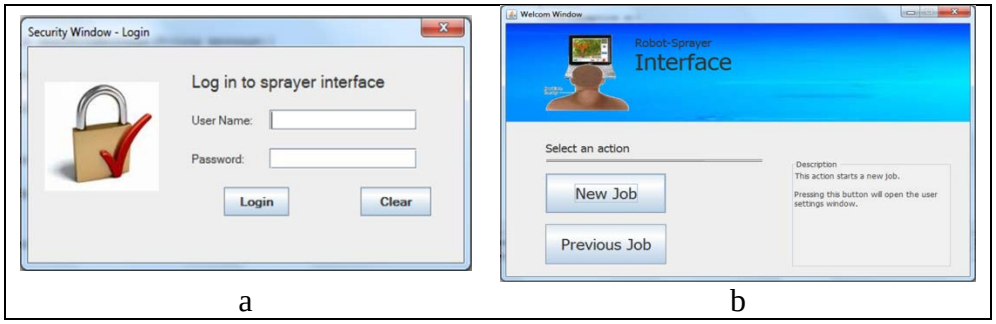
Welcome window. After supplying the correct details on the security window the welcome window appears (Fig. 5.b). The welcome screen contains two push buttons for the user to choose, “New Job” and “Previous Job”. When selecting the “New Job” option, the user is promoted to the “User Settings Window”. When choosing the “Previous Job” the user can choose from previous work settings.

User Settings window. Using this window the user can set different parameters values regarding the spraying process (e.g., number of rows). This

window (Fig. 6) appears only if the user selected “new job” in the welcome window screen. When clicking the “next” button the user is promoted to the “Manual Control Window”.

Manual Control Window. When promoted to this window (Fig. 7) the user has full manual control over all of the robot features: using the front and back cameras the user can view the surroundings, using the navigation arrows or the virtual joystick the user can navigate the robot along the vineyard rows and by clicking the spray function the robot will start spraying. The use of this screen is mainly to navigate the robot to the starting spraying point and for navigation at the end of the automatically spraying process. By clicking the “switch to main” button, the user commands the robot to start the target specific spraying process and the user is promoted to the “Main window”.

Main Window. The main window was designed to allow the user the possibility to help the robot with the task of marking the grape clusters. The grape clusters marking are demonstrated in Fig. 8. The main window was designed to have minimal distractions so that the user could focus on the target marking task. This window includes the emergency stop button which causes complete shutdown of all robotic systems and the “switch to manual control” which promotes the user back to the manual control screen.



a b

Fig. 5. (a) Security window, (b) Welcome window

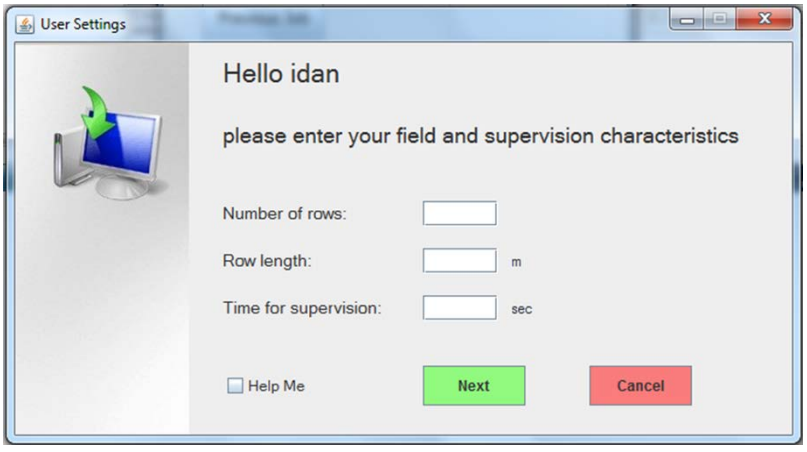


Fig. 6. User settings window

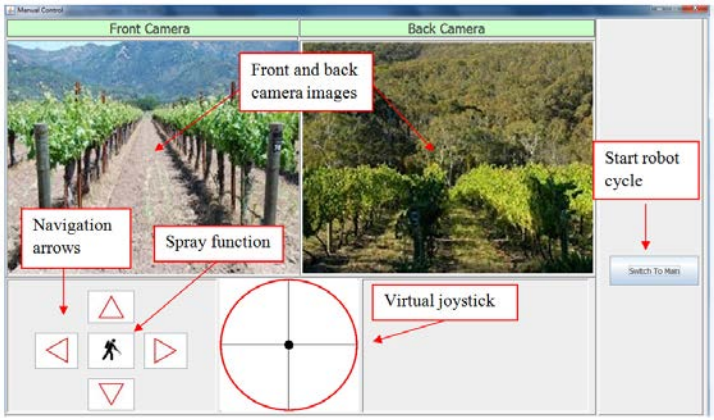


Fig. 7. Manual control window

Fig. 8. Main window

The interface was tested successfully in laboratory experiments. The experiments included functionality testing of all modules (Ben Halevi, 2011). Ongoing research aims to evaluate interface simplicity, friendliness and reliability in field operations (Ben Halevi, 2011).

SUMMARY

The interface presented in the paper was implemented in a vineyard spraying robot. The methodology applied can be extended for other agricultural interface designs. Future work will include extensive experiment in field conditions where the interface compatibility to the user and to the field conditions will be examined in parallel to usability tests.

ACKNOWLEDGMENTS

This research was funded in part the Ben-Gurion University of the Negev Paul Ivanier Center for Robotics Research and Production Management and Rabbi W. Gunther Plaut Chair in Manufacturing Engineering.

REFERENCES

Ben Halevi, I. 2011. Human-robot cooperative interface for vineyard robot sprayer. *Unpublished final project report. Dept. of Industrial Engineering and Management, Ben-Gurion University of the Negev, Beer Sheva, Israel 84105*.

Berenstein, R., and Y. Edan. 2012. Evaluation of marking techniques for a human-robot selective vineyard sprayer. In *International Conference of Agricultural Engineering (CIGR-AgEng)*. Valencia, Spain: Unpublished.

Berenstein, R., O. B. Shahar, A. Shapiro, and Y. Edan. 2010. Grape clusters and foliage detection algorithms for autonomous selective vineyard sprayer. *Intelligent Service Robotics* 3(4):233–243.

Betarbet, R., T. B. Sherer, G. MacKenzie, M. Garcia-Osuna, A. V. Panov, and J. T. Greenamyre. 2000. Chronic systemic pesticide exposure reproduces features of parkinson's disease. *Nature neuroscience* 3(12):1301-1306.

Blackmore, S., H. Have, and S. Fountas. 2001. A specification of behavioural requirements for an autonomous tractor. In *6th International Symposium on Fruit*. Potsdam, Germany.

Cho, S. I., and N. H. Ki. 1999. Autonomous speed sprayer guidance using machine vision and fuzzy logic. *Transactions of the ASAE* 42(4):1137-1144.

Dasgupta, S., C. Meisner, D. Wheeler, K. Xuyen, and N. Thi Lam. 2007. Pesticide poisoning of farm workers–implications of blood test results from vietnam. *International journal of hygiene and environmental health* 210(2):121-132.

Deacon, J. 2009. Model-view-controller (mvc) architecture. *Computer Systems Development*.

Eddleston, M., L. Karalliedde, N. Buckley, R. Fernando, G. Hutchinson, G. Isbister, F. Konradsen, D. Murray, J. C. Piola, and N. Senanayake. 2002. Pesticide poisoning in the developing world—a minimum pesticides list. *Lancet* 360(9340):1163-1167.

Elkabetz, P., Y. Edan, A. Grinstein, and H. Pasternak. 1998. Simulation model for evaluation of site−specific sprayer design. ASAE Paper No. 981013, ASAE St. Joseph, MI 49085

Gil, E., A. Escol, J. R. Rosell, S. Planas, and L. Val. 2007. Variable rate application of plant protection products in vineyard using ultrasonic sensors. *Crop Protection* 26(8):1287-1297.

Goudy, H. J., K. A. Bennett, R. B. Brown, and F. J. Tardif. 2001. Evaluation of site-specific weed management using a direct-injection sprayer. *Weed Science* 49(3):359-366.

Jeon, H. Y., L. F. Tian, and T. Grift. 2005. Development of an individual weed treatment system using a robotic arm. ASAE Paper No. 05-1004, ASAE St. Joseph, MI 49085

Lamm, R., D. Slaughter, and D. Giles. 2002. Precision weed control system for cotton. *Transactions of the ASAE* 45(1):231-238.

Rogan, W. J., and A. Chen. 2005. Health risks and benefits of bis (4-chlorophenyl)-1, 1, 1-trichloroethane (ddt). *Lancet* 366(9487):763-773.

Van Dam, A. 1997. Post-wimp user interfaces. *Communications of the ACM* 40(2):63-67.