Journal of Intelligent and Robotic Systems **8**: 267–284, 1993.

Intelligent Control System Simulation of an Agricultural Robot*

Y. EDAN, B.A. ENGEL, and G.E. MILES
Department of Agricultural Engineering, Purdue University, West Lafayette, IN 47907, U.S.A.

(Received: 27 June 1991; in final form: 28 February 1992)

Abstract. An intelligent control system for an agricultural robot which performs in an uncertain and unstructured environment was modelled as distributed, autonomous computing modules that communicate through globally accessible blackboard structures. The control architecture was implemented for a robotic harvester of melons. A CAD workstation was used to plan, model, simulate and evaluate the robot and gripper motions using 3-D, real-time animation. The intelligent control structure was verified by simulating the dynamic data flow scenarios of melon harvesting. Control algorithms were evaluated on measured melon locations.

Picking time was reduced by 49% by applying the traveling salesman algorithm to define the picking sequence. Picking speeds can be increased by a continuous mode of operation. However, this decreases harvest efficiency. Therefore, an algorithm was developed to attain 100% harvest efficiency by varying the vehicle's forward speed. By comparing different motion control algorithms through animated visual simulation, the best was selected and thereby the performance improved.

Key words. Intelligent control, simulation, agriculture, robotics, melon harvesting, intelligent systems, planning.

1. Introduction

The advent of intelligent machines in agriculture has the potential to raise the quality of fresh produce, lower production costs and reduce the drudgery of manual labour (Sistler, 1987). Robots are perceptive machines that can be programmed to perform a variety of agricultural tasks such as spraying, cultivating, transplanting and harvesting. However, a robot acting in the agricultural environment must contend with an uncertain and unstructured environment that requires development of technologies to solve difficult problems such as

- Mobile operation in a three-dimensional, changing track;
- Random location of targets, i.e., fruit;
- Variability in fruit size and shape;
- Delicate products; and
- Hostile environmental conditions like dust and extreme temperature and humidity.

* Journal Paper No. 13043, Agricultural Experiment Station, Purdue University, W. Lafayette, IN 47907, U.S.A. This research was supported by Grants No. US-1254-87 and US-1682-89 from BARD, the United States-Israel Binational Agricultural Research and Development Fund.

To achieve fast and robust operation in such a complex and dynamic environment, the agricultural robot must be equipped with sensing, scheduling and adaptive planning capabilities. An intelligent sensor-based control system must be structured for efficient planning, coordination and control of these tasks (Kak *et al.*, 1986, Ish-Shalom, 1987; Waldon *et al.*, 1987). The control system must be capable of: monitoring and interpreting data from multiple sensors; transforming the multitude of data to information; planning tasks and controlling task execution while dynamically making conclusions about the environment and reevaluating the intermediate targets as additional data are acquired (Nagdhy *et al.*, 1988; Cox and Gehani, 1989).

The *objective* of this work was to develop an intelligent control system for an agricultural robot. Specific *objectives* were to: define how to organize sensory data; define how information is represented, coordinated and communicated; and develop methods for on-line integration of sensed data with dynamic motion planning and execution. The intelligent control system developed in this work was implemented for robotic melon harvesting. Melons (*Cucumis melo* L., McGlasson and Pratt, 1963) do not ripen uniformly and therefore must be selectively harvested. There is no current machine available for harvesting melons (Lenker, 1984). Melons were selected as an example for using robotics with other delicate and high-value horticultural products.

The data flow and plan execution must be verified before implementing it on actual hardware to confirm that the control architecture performs effectively and to ensure that the algorithms and computer programs execute as intended. Since the intelligent control architecture involves functional and logical flow of data, which are of qualitative nature, it could not be verified using numerical algorithms and thus, conventional simulation methods were not suitable for model verification. In the agricultural environment, simulation is essential because of the variability of crop conditions (Miles and Tsai, 1987). Experiments are difficult to reproduce since crop conditions change with time and vary spatially in the field. Performance differences caused by changes in the control algorithms are often impossible to distinguish from those caused by changes in crop conditions.

In this work, recent developments in graphic simulators that have been developed to design robotic systems (Nof, 1991) were applied to verify the intelligent control system. Using computer graphics, operations were programmed, animated and observed through a dynamic, realistic, three dimensional visualization of the robot mechanisms. Incorporating environmental changes into the simulation model was necessary to verify dynamic planning and execution. Algorithms for real-time sensory interpretation are beyond the scope of this study. On-line data acquisition and processing are prerequisites to implementation of a sensory-based control system and the necessary hardware and software are commercially available. This work develops a sensor-based control system and ensures that the proposed intelligent control system performs correctly and consistently for sensory data supplied on-line.

The intelligent controller model developed for agricultural robots is presented in the following section. The implementation of the intelligent control model for

robotic melon harvesting is presented in the third section which describes the model, software, database utilized and implementation. This chapter also presents methods for verifying the control model prior to implementation in the realworld including modelling of dynamic environmental changes. Planning algorithms are next described and include sequence planning, motion planning and local guidance algorithms. The last section includes the summary, conclusions and future trends.

2. Control System Architecture

A robot performing in the uncertain and unstructured agricultural domain must be equipped with sensors. Multiple and different types of sensors are required to extract accurate information on the robot's surroundings because of the variability and disturbances in the agricultural environment and limitations intrinsic to any kind of sensor. This multitude of data must be sensed and processed in real-time to form information and knowledge such as existence and location of the fruit. Based on this knowledge, tasks and motions must be planned, executed and modified using on-line sensory information. The major architectures for real-time control of complex systems are: hierarchical control, message based and blackboard systems (Saridis, 1983, 1988; Brooks, 1986; Elfes, 1986; Nii, 1987; Meystel, 1988). The intelligent controller for the agricultural robot was based on the blackboard model since it provides a flexible framework for complex and unstructured problems (Nii, 1987; Barrett and Jones, 1989). A blackboard is a globally accessible database where different processes that are cooperating toward the solution of a certain problem can share data and results, suggest hypotheses to be examined, communicate the state of their own computation, etc. (Nii, 1987). The blackboard model uses opportunistic reasoning as its problem solving model, i.e., the knowledge module applied is determined dynamically, one step at a time, rather than in a fixed and systematic sequence (Engel *et al.*, 1990). Pieces of knowledge are applied in the most appropriate manner and at the most opportune time. This is important for robust operation in the dynamic agricultural environment where for example, changes in light conditions because of clouds, or shades require different image processing routines; and fruit distribution variations imply activating different motion control algorithms. For each part of the problem and at each stage of the solution formation, the best knowledge representation and solution strategy can be selected. Moreover, qualitative (rule-base, heuristic) and quantitative methods (numerical solutions) can be coupled (Engelmore and Morgan, 1988). Therefore, various types of sensory data and diverse information can be integrated.

To reduce complexity, the intelligent controller of the agricultural robot was divided into two hierarchies: a high-level planning blackboard and a low-level control blackboard (Figure 1): Sensing, task planning, and action were distributed between independent modules that communicate asynchronously over globally accessible blackboards at the highest possible level, i.e., each module transfers only essential information.

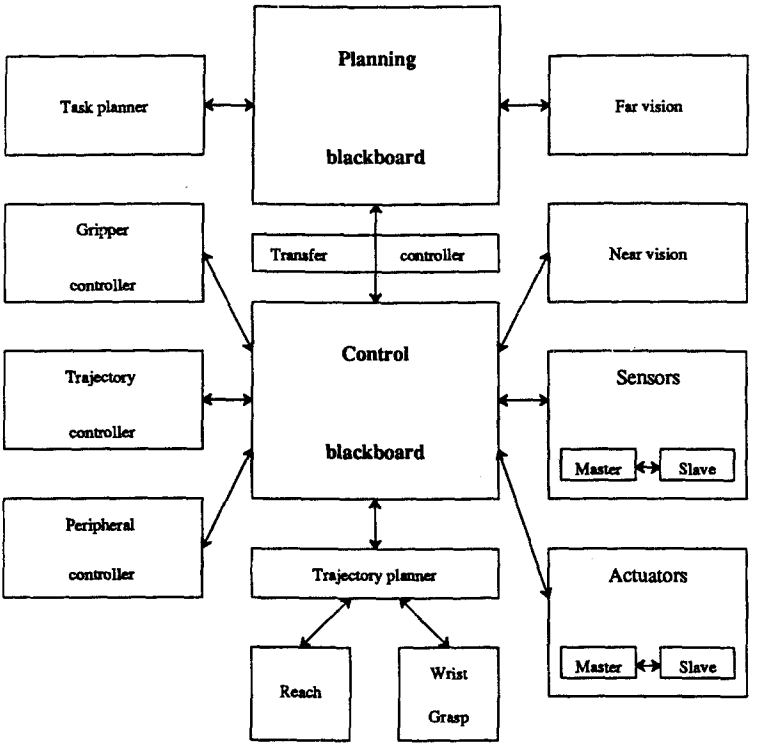

Fig. 1. Intelligent control system for an agricultural robot.

2.1. VISION MODULE

Vision provides the most important source of information about the robot's environment. Because of complexity, vision is independent and distinguished from other sensory systems. In general, vision provides navigation and object recognition and classification capabilities that involve two distinct processes: global path planning and local guidance. A hierarchy of sensors is introduced for the planning (far-vision) and control (near-vision) levels. Each system updates a blackboard and is divided into two levels: the sensor control (i.e., the data acquisition algorithms, the executable programs that obtain the data), and the processing level (i.e., the image processing algorithms that transform the data into information). The input to these modules are the acquired images, the output is the extracted information. Simulation of the vision module is achieved either by execution of the actual image processing routines derived for the specific application or by a process that presents the required information at distinct time intervals. The time intervals should represent typical image processing times. To ensure that data flow is consistent and independent of the image-processing duration, these time intervals should be varied between consecutive loops. Different environmental conditions should be simulated by executing a process that changes its status inconsistently, i.e., at random intervals to randomized conditions. The vision module should select the preferable image

processing algorithm based on the current environment conditions. Thus, the execution of different algorithms according to different conditions can be tested.

Far-vision: The machine vision sensor scans scenes ahead of the robot's current position. For autonomous mobile vehicles, it provides guidance directions and information about obstacles. For robotic harvesters, this system recognizes and classifies object information such as position, orientation, visual texture, colour, shape and size of fruit. A certainty factor indicates the degree of confidence in the validity of the information.

Near-vision: The machine vision sensor is mounted on the end effector of the robot and provides visual feedback for motion control and updates of exact position and orientation of the target. Simulation of this module should incorporate changes of the information supplied by the far-vision system to verify the adaptive planning capabilities of the intelligent control system.

2.2. TASK PLANNER

The task planner defines the tasks to be performed, divides them into subtasks and determines the appropriate sequence. It contains global strategies that optimize the planned sequence of tasks to minimize performance criteria such as the time to pick all the fruit. The input to this module is the information supplied by the sensory systems such as locations of fruit to be picked, orientation and position of the robot end-effector, etc. Simulation of this module is achieved by implementing the actual task planning algorithm.

2.3. TRANSFER CONTROLLER

This module transfers information from the planning blackboard to the control blackboard. In addition, it assigns task priorities based on local strategies that may modify the plan according to new information obtained from real-time processing of sensed data. The transfer controller provides error recovery and replanning in case of system failure, unexpected events or alarm occurrences. The input to this module is the sequence of tasks to be performed and sensory data. The output is the next task to be performed. Transfer of information and local strategies should be simulated as intended to be implemented. Alarm occurrences should be simulated by a process that occurs at random time intervals simulating unpredictable conditions.

2.4. TRAJECTORY PLANNER

This planner generates trajectories to move the robot arm from its current position to the intended target. Strategies to determine where the arm should move, when it should stop and wait for new sensory data, and how it should revise its original plans have been developed. This level is horizontally decomposed into concurrent

processes for planning the reaching and wrist/grasping motions. Reach planning provides the coarse coordinates of the trajectory. Near-vision and sensor data provide information to plan wrist orientation and grasping motions. The inputs are the robot's current position and the location of the next target. The outputs are then positions, velocities and accelerations of the robot's joints that form the trajectory that the robot's arm should follow. This module is simulated by a process that executes the intended trajectory planner algorithm.

2.5. TRAJECTORY CONTROLLER

The trajectory controller senses the position of each actuator and signals each motor to comply with the planned path. Deviations and position data are communicated to the trajectory controller. The inputs to this module are the current location of the robot and the target. The outputs are control signals to the robots actuators. Since the trajectory controller is usually incorporated into the robotic system using closed-loop control, there is no need to simulate this module for verification of the intelligent control system. Thus, only the actual execution of the trajectory should be simulated. Verification of the dynamic control of the robot manipulator is beyond the scope of this work.

2.6. GRIPPER CONTROLLER

The gripper controller modifies the wrist and grasp motions based on proximity information derived from near vision and other sensors. It communicates fine corrections to the trajectory controller to accurately reach the target with the required orientation. The inputs to this module are the current location and orientation of the gripper and the target. The outputs are control signals to the robot and gripper's actuators. The gripper controller should be simulated by implementing the actual algorithms.

2.7. SENSORS AND ACTUATORS MODULES

These modules control the necessary sensors and actuators. Each module consists of two processes that communicate directly through messages: a master and a slave. The master retrieves necessary information from the blackboard, distributes it to its slave and updates relevant information provided by the slave. The slave monitors the sensors and controls the actuators. Since time is critical this is a stand-alone process that may reside in a different processor or include special purpose electronic circuits or mechanics. The input for the sensor modules are the environmental data and conditions that define when to execute the data acquisition programs such as time constraints and interrupts. The output is the derived information such as speed, position, steering angle of the vehicle or location and ripeness of fruit. The input for the actuator modules are the necessary control

commands, e.g., move to location X, Y and the conditions on when to move. The outputs are control commands, e.g., move the robot arm and steer the vehicle. These modules should be simulated in a similar manner as the aforementioned vision module and gripper controller.

2.8. PERIPHERAL CONTROLLERS

These modules control complex high-level devices such as additional robots performing similar tasks in parallel or performing secondary tasks.

3. Robotic Melon Harvesting

3.1. MODEL

The robotic melon harvester model is based on a prototype field crops robotic machine (Benady *et al.*, 1991) and consists of a robot connected to a tractor that advances along the row in two different modes: step mode and continuous mode. In the step mode, the tractor advances along the row and stops at each grouping of fruit. Then the robot arm approaches each fruit and picks the ripe ones. Once all ripe fruit are picked, the tractor advances. In the continuous mode, the robot arm picks the fruit while the tractor is moving. The far-vision system is mounted ahead of the current position of the vehicle and robot and determines melon locations. A fan directed on the foliage exposes the melons that may be hidden by the canopy. A combined brightness and texture algorithm is used to detect the melons (Cardenas-Weber *et al.*, 1991). The near-vision system is mounted on the robot end effector and provides local guidance of the arm to the next fruit to be picked. Algorithms for supplying 3-D information of the scene are being developed (Benady *et al.*, 1991). A gripper (Wolf *et al.*, 1990) is attached to the vertical axis. Contact sensors are mounted on the gripper to avoid collision with the ground. The gripper is attached to the robot in a flexible joint to absorb side loads caused by horizontal motion.

3.2. SOFTWARE DESCRIPTION

Animated motions of the robot and gripper were displayed using IGRIP (Interactive Graphics Robot Instruction Program version 1.6, Deneb Robotics, IGRIP, 1988) on a Silicon Graphics, Personal Iris 4D/20 workstation (Silicon Graphics, 1988).* The IGRIP software is a computer graphics based package for robotic workcell layout, simulation and offline programming. A workcell is composed of devices and robots, positioned relative to each other. Devices in the workcell are pro-

* Mention of a particular manufacturer and products are for informational purposes only and does not imply endorsement by the authors or Purdue University.

grammed using the high level GSL language (Graphics Simulation Language, Deneb Robotics, GSL, 1988). GSL provides the ability to construct programs for individual devices in a simulation to govern their actions and behavior. Several devices can be simulated simultaneously and are synchronized through Input/Output signaling.

3.3. DATABASE

To evaluate the performance of the control structure based on realistic input, exact field coordinates of melons, cultivar 'Superstar' (*Cucumis melo*, L. var. *reticulatus* Ser., McGlasson and Pratt, 1963) were collected on 859 melons in 29 rows at the Southwest Purdue Agricultural Center, 7 km north of Vincennes, Indiana, in the summer of 1989 (Edan and Miles, 1991). These data served as the input to all simulations performed. A statistical study (Edan, 1990) indicated that the average distance between adjacent fruit along the row (X) was 34 cm with a standard deviation of 10 cm. The average distance between fruit across the row (Y) was 23.6 cm with a standard deviation of 14.2 cm. The histogram of distance between fruit clusters best-fits the Gamma distribution curve. This distribution was used to generate additional simulation datasets corresponding to common horticultural practices. Melon locations were obtained by changing plant distances between 25 and 125 cm which are common horticultural practices (Edan, 1990). The number of fruit per plant were computed using the equations derived by Bhella (1984). For each planting distance, three different random seed numbers were used to generate the data using the inverse transformation method (Pritsker, 1986).

3.4. IMPLEMENTATION

Each module of the intelligent controller was connected to a different device in the IGRIP workcell and had a separate GSL program (Figure 2). All programs were executed concurrently. All information was exchanged through blackboards that were implemented by globally accessible data files. Lists of observations from the sensory systems arrive at the blackboard asynchronously. All references to the blackboard are part of a transaction. Each module keeps track of its current status by storing time flags in a dedicated control file. Data obtained at different times are saved in sequentially numbered files. To ensure access of relevant data, each module checks the time flag in its own control file and the control file of the module responsible for deriving the required data before accessing data.

Initially, the far-vision system was implemented by executing the image processing routines that were developed for melon detection (Cardenas-Weber, 1991). These algorithms determine the location and size of fruit and a certainty factor for detection. The image processing strategy consists of two main functions: an analysis operation and a knowledge directed evaluation (Cardenas-Weber *et al.*, 1991). The analysis operation provides the positions of the melons in the area of interest based

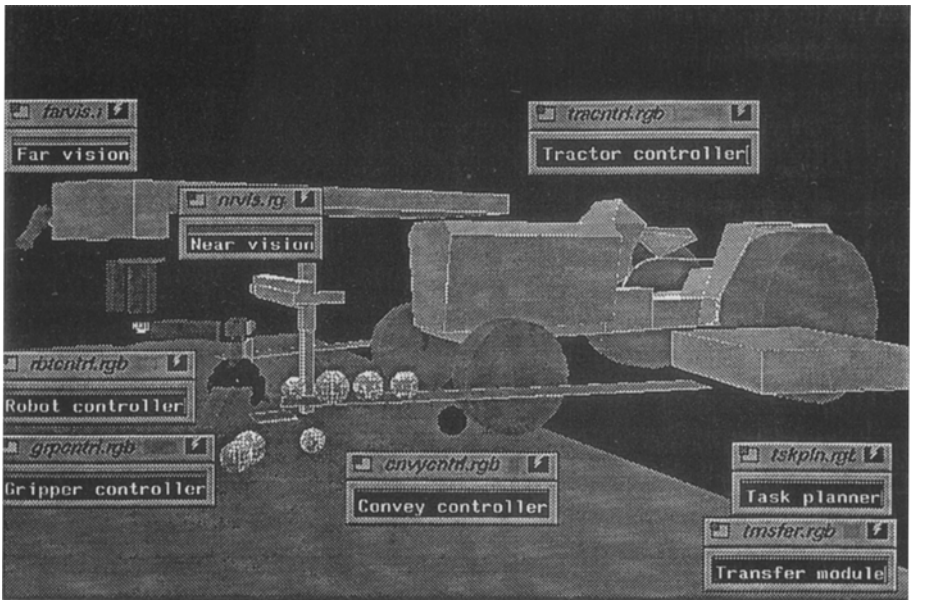

Fig. 2. Description of the IGRIP simulated modules.

on object brightness, size, area, shape, distance between objects and distance between plants. However, the disparities of the information (inconsistencies and contradicting values) requires a knowledge-based strategy to improve the detection correctness. These knowledge rules represent and utilize information about the application domain and specific situations such as characterizing melons under different brightness conditions and occlusion. An example for such a rule is comparing the detected object size (width, height, radius) with the reference values for maximum, minimum and average melon size.

However, since these algorithms were developed for feasibility and not optimized for real-time application, the actual execution of these algorithms was time consuming (i.e., all robot motions were stopped while waiting for the image processing routines to be executed which interfered with the continuous and dynamic flow of motions). Thus, they were deleted from the simulation process and were *simulated* by adding a delay loop into programs called by the camera device. To validate that the data flow between computing modules does not depend on the vision rate, the delay between consecutive loops was varied. Fruit locations are read from data files that contain the measured melon coordinates. The real-time issues of vision data interpretation are being approached by implementing these algorithms on real-time pipeline image processing architecture which increases the processing speeds (Benady *et al.*, 1991). More sophisticated algorithms that adapt to changing environmental conditions have not been developed yet, however their simulation was demonstrated: A random number was entered into a file that represented the environment. Different image processing routines

simulated by different delay loops were executed according to this number.

Once the *simulated* vision program is completed, melons are displayed in the workcell:

- a green melon indicates that it is unripe (small or green as detected by the far-vision system);
- an orange melon indicates that it is ripe (and has to be further investigated by the far-vision system to determine exact ripeness stage).

Based on these locations, the task planner determines the stopping point of the vehicle and the sequence of fruit to be picked. The transfer module transfers information from the planning blackboard to the control blackboard. Based on this list of fruit and new fruit detected by the near-vision system, it defines the next fruit to be picked. The robot controller guides the arm to this fruit. Once the robot arm is located above a fruit to be picked, it starts moving and is guided by the near-vision system that also determines the ripeness stage. Based on local guidance heuristics, the arm is positioned above the fruit and waits for a signal from the near-vision system. Once the fruit has been detected by the near-vision system and depending on its ripeness stage, the robot either picks the fruit or restarts the picking cycle with the next fruit. The near-vision system is implemented by simulating a process that generates various random conditions to simulate the following dynamic conditions:

- Detecting the fruit at the location indicated by the far-vision system.
- Random changes in the fruit locations (simulates inaccuracies in the far-vision system detection).
- Displaying *new* fruit into the workcell that were not identified by the far-vision system (indicated by changing the melon's colour to white).
- Deleting existing fruit from the workcell (that were falsely detected by the far-vision system – this is displayed by changing the melon's colour to purple and then its display to invisible).

After the arm reaches the target, a gripper is activated and picks the fruit up from the ground, detaches it from the vine and transfers it to a conveying system. This time is denoted as the picking time. Once all the fruit in the current group have been picked, the vehicle advances to the median of the next cluster of fruit that have been detected. This process continues until all fruit along the row have been picked. The sequence of animated motions shown in Figure 3 is

(a) the robot arm approaches the fruit, picks it up, transfers the picked fruit to the nearby intermediate conveyor;
(b) the robot arm places the fruit on the intermediate conveyor;
(c) the robot arm approaches the next fruit while the intermediate conveyor conveys the fruit to the main conveying system; and
(d) the robot arm picks the next fruit.

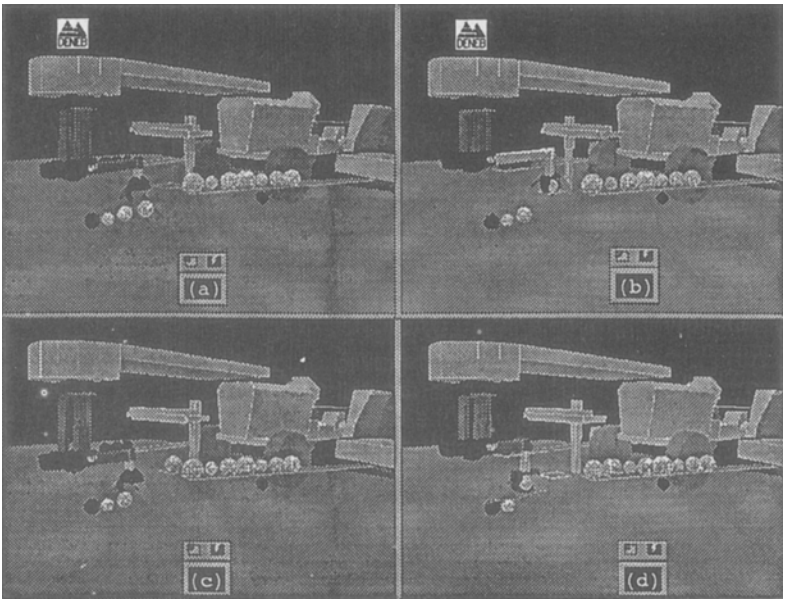

Fig. 3. A sequence of the robot's motions.

3.5. PERFORMANCE EVALUATION

The logical flow of the data was ensured for various scenarios by varying the following parameters:

- maximum velocity and acceleration of the robot actuators (varied by changing robot parameters in the IGRIP workcell: velocities of 500, 1000, and 2000 cm/sec and accelerations of 0.5, 1.0, and 2.0 g were simulated);
- sensor execution time (adding a delay between 1 and 500 msec in the vision program);
- planning execution time (could only be increased; a delay loop was added into the task planner program);
- number of ripe fruit;
- distribution of fruit;
- environmental conditions; and
- detection certainty factors.

A different test case was produced for each simulation, by assigning random numbers (generated from a uniform distribution) to the number of ripe fruit and detection certainty factors. Percentage of ripe fruit versus unripe fruit were varied by assigning different thresholds to the random numbers that were assigned to simulate these parameters. Fruit that received a number larger than the setup threshold were simulated to be ripe, others unripe. Distribution of fruit was varied by changing the input data to consist both of measured field data and stochastically generated data (corresponding to different planting distances). Thus, for each simulation test, a

different row of melon coordinates was simulated. An additional random number was assigned to each fruit at the ripeness detection stage and simulated dynamic changes in the sensory information provided by the near-vision system, (e.g., changes in fruit location, and detection of fruit). Simulation of different environment conditions is achieved by assigning a random number at random times to a file. The vision module checks this file to decide which algorithm it should execute. Thus, the adaptive planning algorithms were evaluated for simulated real-time changes in the sensory data.

3.6. RESULTS

The control architecture was verified by observing the motions of the robot on the screen and ensuring the following criteria:

- All ripe fruit were picked and conveyed to the correct conveyor according to their ripeness stage; and
- All unripe fruit were left in the field.

Through an interactive process of simulating and visualizing the motions, all algorithms were verified and it was ascertained that each submodule has the necessary information at the right time for all conditions and the programs performed correctly. This ensured that all program codes were tested before being implemented in the real-world robot. Since the performance criteria were met for all simulated parameters, for all field conditions tested and for dynamic changes in the sensory data, the implemented control architecture is considered verified.

4. Planning Algorithms

4.1. SEQUENCE PLANNING ALGORITHM

To increase the system's throughput, the sequence of robot motions before the beginning of the picking process was preplanned. The minimum-time sequence to pick all the fruit was obtained by applying the Traveling Salesman Problem (TSP) algorithm (Edan, 1990). The TSP involves routing a salesman through a sequence of known locations so as to minimize the time or any other well defined function (Reingold *et al.*, 1983). The cost function for the TSP was calculated as in Drezner and Nof (1984) to be the distance to move the empty arm from the place on the conveyor to the next fruit. Whenever the robot arm was ready for the next target, the search was halted and the current best solution defined the next sequence of fruit to be harvested. Thus, at each point in time the best solution was obtained. The algorithm was evaluated by simulations performed for a Cartesian robot with actuator speeds of 1 cm/sec on locations of measured melon data. Figure 4 shows the difference between the minimum-time traveling salesman solution (optimum) and a solution received from the first evaluated sequence (regular). The total time was reduced by 49.44%.

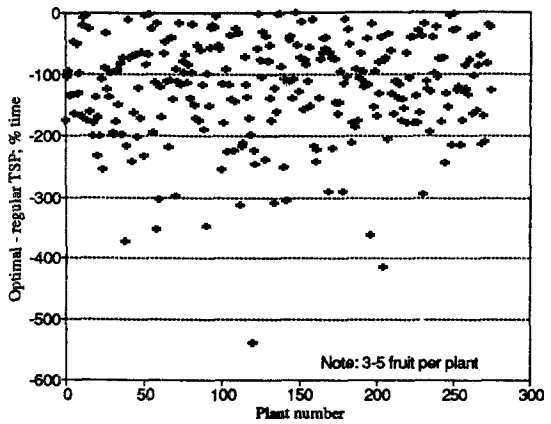

Fig. 4. Percent cycle time difference between optimum algorithm and non optimal TSP.

4.2. MOTION PLANNING ALGORITHMS

Step mode: Integration of qualitative and quantitative methods was demonstrated by implementing the following heuristics and optimization algorithms to define the tractor's motions along the growing bed in the step mode: For each fruit cluster, the median location is calculated and the tractor is stopped to align the robot's base with the median of the fruit. If the distance between sequential fruit is less than 100 cm and if the number of ripe fruit is less than 5, the stopping point is the median of these fruit. Otherwise, a cluster analysis was performed on all available data to determine the number of fruit clusters. The cluster analysis algorithm (Pal and Duda, 1987) was implemented by iterative optimization to minimize the sum-of-squared-error (Edan, 1990). The median is derived for each cluster.

The heuristic planning algorithms were implemented by rules in a CLIPS (1989) program that is called from the GSL program. CLIPS is a rule-based, forward-chaining expert system shell and was selected since it is highly portable, easily integrated with external software and easily embedded into other systems. Optimization algorithms were implemented by integrating CLIPS with external functions written in the C programming language.

Continuous mode: To improve the robot harvester's performance, the fruit should be picked while the vehicle advances along the row. However, if the robot is slow and the tractor's forward speed is high and constant, the harvest efficiency is reduced since many fruit are missed by the robot. The continuous mode algorithm was evaluated for a Cartesian robot with actuator speeds of 100 cm/sec on the measured field data. As the forward speed and picking time increase, fewer fruit are picked (Figure 5). Increasing the forward speed from 22 to 66 cm/sec decreases the percentage of fruit picked from 100 to 30 for a picking time of 0.5 sec and from 45 to 1 for a picking time of 2.0 sec.

To increase field capacity (i.e., acres/hour) and attain a high percentage of picked fruit, an algorithm was developed to vary the vehicle's forward speed so that the

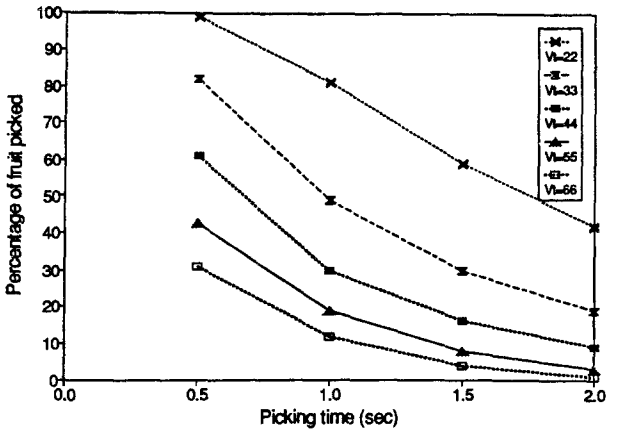

Fig. 5. Effects of picking time and forward velocity on percentage of fruit picked.

robot has just enough time to pick all the fruit. According to the number of fruit in the next cluster, the vehicle speed changes: when fruit density is low the forward speed is increased and when many fruit are clustered together the vehicle decelerates, giving the robot more time.

Figure 6 presents the effect of picking time on the total time to harvest a row for constant and variable velocities of 11 and 66 cm/sec. For constant velocities, the total time decreases as the picking time increases since fewer fruit are picked. For variable velocities all fruit are successfully picked and thus, as the picking time increases the total time increases.

Figure 7 shows the effect of the picking time on cycle times for constant and variable velocities of 11 and 66 cm/sec. The time per fruit increases as the forward speed decreases and the picking time increases. The variable velocity mode is faster than the constant velocity mode by 3.3%.

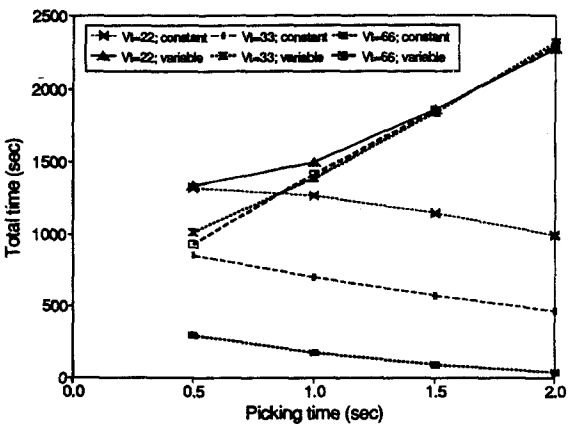

Fig. 6. Effect of picking time on total time for variable and constant velocities.

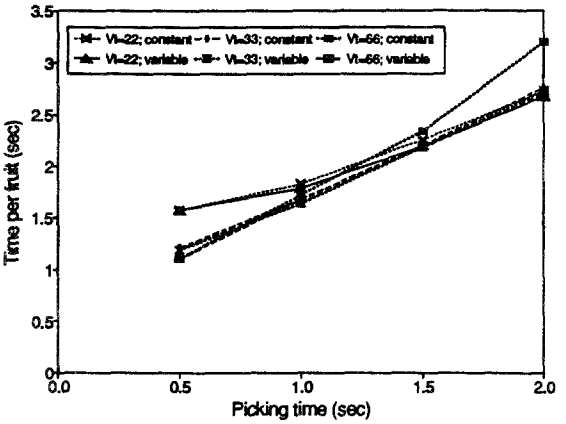

Fig. 7. Effect of picking time on cycle times for variables and constant velocities.

4.3. LOCAL GUIDANCE ALGORITHMS

The flowchart of the robots motions is given in Figure 8: The robot arm moves down to a point 35 cm above the next fruit to be picked and waits for information from the near-vision system. If the far-vision system indicated detection with high accuracy, the robot arm moves down 30 cm and waits for ripeness information from the near-vision system. If lower certainty has been indicated, the arm moves down only 10 cm, to a distance of 25 cm from the fruit, to provide the near-vision system

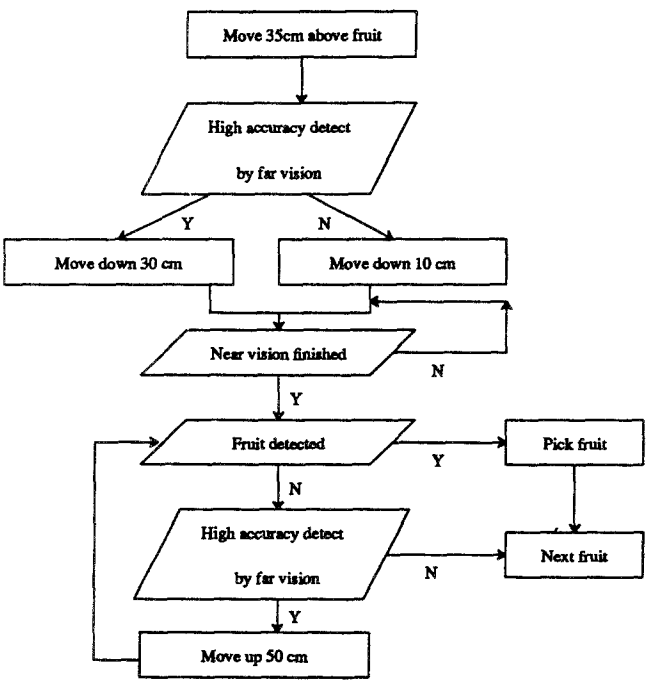

Fig. 8. Flowchart of the robot's motions.

with a larger field of view. False detections of the far-vision system were simulated by changing the fruit to invisible, i.e., the fruit *disappeared* from the workcell. If the fruit *disappears* while the arm approaches the fruit and the accuracy of detection of the far-vision system was high, the arm stops and moves up and asks the near-vision system to recheck, and waits for an acknowledge signal for continuation. If the fruit *disappears* while the arm moves toward it and the accuracy of detection was low, the arm proceeds to pick the next one. The execution of these modified plans are visualized on the screen, i.e., actions of the robot arm change according to the simulated dynamic environmental conditions.

5. Summary, Conclusions, and Future Development

An intelligent controller capable of interfacing sensors and control devices has been designed, simulated and verified for a robot acting in the unstructured and uncertain agricultural environment. The agricultural robot control system consists of a hierarchical control architecture that activates through multiple blackboards. Each level consists of independent modules that communicate through their corresponding blackboard. The blackboard was implemented by sharing data through files. The blackboard provided a flexible and modular structure that facilitated modifications and addition of modules and coupling of different problem-solving methods. This is essential for the complex agricultural domain. All stages of the data flow have been simulated: starting with input from the sensors, through transformation to information and plans, and to final execution of tasks and control. Dynamic changes of sensory data have been simulated, and algorithms, heuristics and optimization methods incorporated into the planning and execution of the motions for the robotic melon harvester. The intelligent control structure proved to provide efficient and robust operation of the agricultural robot for a variety of simulated environmental conditions and for a range of sensory processing, task planning and motion control execution durations.

By observing the robot's motions for various sets of system parameters and field conditions, the control architecture was verified and the logical flow of the data proved to be correct and consistent for a variety of conditions prior to implementation in the realworld robot. Animated, visual simulation provided an important and powerful tool for verifying the control model, debugging all sensor, planning and robot control computer programs and evaluating motion control algorithms. Simulations of dynamic changes of sensory data was essential to ensure robust operation.

Evaluation of planning algorithms on measured melon locations indicated that the traveling salesman algorithm decreases the cycle times required to pick the fruit by finding a minimum-time sequence by 49% and therefore should be implemented. A continuous mode of operation decreases cycle times. Variable speed control improves performance significantly as compared to a constant velocity mode where many fruit are missed at high forward speeds. By controlling the velocity, all fruit are successfully picked and the overall speed is increased.

The evaluation of these algorithms prior to prototype testing ensures improved performance and reduces the amount of experimental research. Prior to implementation on an operational system, the intelligent controller must be extended and implemented on a real-time system. Real-time sensory interpretation must be achieved by utilizing dedicated custom-made hardware. Multiple processors should be incorporated to enable concurrent execution of sensing, planning and control. However, the blackboard implementation on a distributed computing system which consists of multiple processors is not straightforward and must be developed (Benady *et al.*, 1991). Since time is critical for real-time control, the blackboard should be implemented by shared memory. Interprocess communication can be achieved using semaphores. Due to associated memory and computational time, expenses related with the blackboard the final implemented structure should be reanalyzed and optimized.

In this paper, a structure to organize the necessary multitude of sensors has been proposed, modeled and simulated. However, these sensors must be defined, i.e., what kind of sensors and how many of each type are required to provide adequate information. Once the sensors have been defined, strategies to decide when and what sensors should be deployed must be formulated. Moreover, algorithms must be developed to fuse the information available from the diverse sensors. The development of the intelligent controller structure presented in this paper is an essential prerequisite to development of a mobile, autonomous field robotic harvester which is currently under development (Benady *et al.*, 1991).

References

Barrett, J.R. and Jones, D.D., 1989, Knowledge engineering in agriculture. An ASAE Monograph No. 8, ASAE, St. Joseph, Michigan 49085.

Bhella, H.S., 1984, Response of muskmelon to within-row plant spacing, *Proc. Indiana Academy of Science* **94**, 99–104.

Benady, M., Edan, Y., Hetzroni, A., and Miles, G.E., 1991, Design of a field crops robotic machine. Paper No. 91-7028, American Society of Agricultural Engineers, St. Joseph, MI.

Brooks, R.A., 1986, A robust layered control system for a mobile robot, *IEEE J. Robotics Automat.* **2**(1), 14–23.

Cardenas-Weber, M., Hetzroni, A., and Miles, G.E., 1991, Machine vision to locate melons and guide robotic harvesting, Paper No. 91-7006. American Society of Agricultural Engineers, St. Joseph, MI.

Cardenas-Weber, M., 1991, Design of a machine vision system for a robotic harvester of melons, Ph.D. Thesis, Purdue University, West Lafayette, IN.

CLIPS, 1989, *Reference Manual*, Version 4.3, Artificial Intelligence Section, L.B. Johnson Space Center.

Cox, I.J. and Gehanni, N.H., 1989, Concurrent programming in robotics, *Internat. J. Robotics Res.* **8**(2), 3–16.

Drezner, Z. and Nof, S.Y., 1984, On optimizing bin picking and insertion plans for assembly robots, *IIE Trans.* **16**(3), 262–270.

Edan, Y. and Miles, G.E., 1989, Animated, visual simulation of robotic melon harvesting, Paper No. 89-7612. American Society of Agricultural Engineers, St. Joseph, MI.

Edan, Y., 1990, Control and design of an intelligent agricultural robot, Ph.D. Thesis, Purdue University, West Lafayette, IN.

Edan, Y. and Miles, G.E., 1991, Design of a robotic melon harvester, Paper No. 91-7029. American Society of Agricultural Engineers, St. Joseph, MI.

Elfes, A., 1986, A distributed control architecture for an autonomous mobile robot. *Internat. J. Artificial Intelligence* **1**(2), 99–108.

Engel, B.A., Beasley, D.B., and Barrett, J.R., 1990, Integrating multiple knowledge sources, *Trans. ASAE* **33**(4), 1371–1376.

Engelmore, R. and Morgan, T., 1988, *Blackboard systems*, Addison Wesley, Wokingham, England.

GSL Reference Manual, 1988, Deneb Robotics Inc., Troy, MI.

IGRIP Simulation System Ver 1.6, 1988, Deneb Robotics Inc., Troy, MI.

Ish-Shalom, J., 1987, Task level specification of a robot control, *IEEE Internat. Symp. Intelligent Control* pp. 196–201.

Kak, A.C., Boyer, K.L., Chen, C.H., Safanek, R.J. and Yang, H.S., 1986, A knowledge-based robotic assembly cell, *IEEE Expert*, Spring; pp. 63–83.

Lenker, D.H., 1984, Factors limiting the harvest mechanization of some major vegetable crops in the U.S.A., In *Fruit, Nut, Vegetable Harvesting Mechanization Symposium*, pp. 29–38.

McGlasson, W.B. and Pratt, H.K., 1963, Fruit-set patterns and fruit growth in cantaloupe, *Amer. Soc. Horticultural Sci.* **83**, 495–505.

Meystel, A., 1988, Intelligent control in robotics, *J. Robotic Systems* **5**(4), 269–308.

Miles, G.E. and Tsai, Y., 1987, Combine systems engineering by simulation, *Trans. ASAE* **30**(5), 1277–1281.

Nagdhy, F., Wai, C.K., and Nagdhy, G., 1988, Multiprocessing control of robotic systems, *IEEE Internat. Conf. Robotics and Automation*, pp. 975–977.

Nii, H.P., 1987, Blackboard systems: the blackboard model of problem solving and the evolution of blackboard architectures, *AI Magazine* **7**(3), 38–53.

Nof, S.Y., 1991, Industrial Robotics. In G. Salvendy (ed.), *Handbook of IE*, Wiley, New York, Chapter 16.

O'Brien, M. and Zahara, M., 1971, Mechanical harvest of melons, *Trans. ASAE* **14**(5), 883–885.

Pal, S.K. and Duda, M.D.K., 1987, *Fuzzy mathematical approach to pattern recognition*, Wiley, New Delhi, India.

Pritsker, A.B., 1986, *Introduction to simulation and Slam II*, Wiley, New York.

Reingold, J.M., Nievergelt, J., and Deo, N., 1983, *Combinatorial Algorithms: Theory and Practice*, Prentice-Hall, Englewood Cliffs, NJ.

Saridis, G.N., 1983, Intelligent robotic control, *IEEE Trans. Autom. Control* **28**(5), 547–557.

Saridis, G.N., 1988, Knowledge implementation: structures of intelligent control systems, *J. Robotic Systems* **5**(4), 254–268.

Silicon Graphics, 1988, *Silicon Graphics Reference Manual*, Silicon Graphics.

Sistler, F.E., 1987, Robotics and intelligent machines in agriculture, *IEEE J. Robotics and Automat.* **3**(1), 3–6.

Waldon, S., Gaw, D., and Meystel, A., 1987, Updating and organizing world knowledge for an autonomous control system, *IEEE Internat. Symp. Intelligent Control*, pp. 423–430.

Wolf, I., Bar-Or, J., Edan, Y., and Peiper, U.M., 1990, Developing grippers for a melon harvesting robot, Paper No. 90-7504, American Society of Agricultural Engineers, St. Joseph, MI.