BARD

FINAL REPORT

PROJECT NO. IS - 1682 - 89C

הספרייה המרכזית
למדעי החקלאות
בית-דגן

Robotic Systems for Selective Harvesting: Integration and Prototype Tests

G.E. Miles, U.M. Peiper, I. Wolf, T. Flash

1994

630.72
BAR/MIL

ROBOTIC SYSTEMS FOR SELECTIVE HARVESTING: INTEGRATION AND PROTOTYPE TESTS

BARD Project No. 1682-89C
Final Report

Cooperating Institutions

Purdue University
West Lafayette, Indiana

Agricultural Research Organization
Bet Dagan, Israel

Weizmann Institute of Science
Rehovot, Israel

DATE: 1 April 1994

BARD
P.O. Box 6
Bet Dagan, ISRAEL

BARD Project No. US-1682-89C

ROBOTIC SYSTEMS FOR SELECTIVE HARVESTING: INTEGRATION AND PROTOTYPE TESTS

Investigators

Gaines E. Miles
Meny Benady
Amots Hetzroni
M. Cardenas-Weber
Uri M. Peiper
Yael Edan
I. Wolf
J. Grinspun
Y. Dobrusin
Tamar Flash
V. Rogozin

Institutions

Purdue University
Purdue University
Purdue University
Purdue University
Agricultural Research Organization
Agricultural Research Organization
Agricultural Research Organization
Agricultural Research Organization
Agricultural Research Organization
Weizmann Institute of Science
Weizmann Institute of Science

Project Start Date: 6 September 1990

Termination Date: 31 December 1993

FINAL REPORT

Gaines E. Miles, Prof
Agricultural Engineering
Purdue University

Yael Edan
Senior Research Scientist
ARO - Volcani Institute

Division of Sponsored Programs
Purdue Research Foundation
Purdue University

630.72
BAR/GAI
2nd copy

ROBOTIC SYSTEMS FOR SELECTIVE HARVESTING: INTEGRATION AND PROTOTYPE TESTS

Abstract

Robotic technologies for selective harvesting of head crops has been developed and a prototype machine has been field tested with muskmelons. Technologies which permit individual fruit to be identified and selectively harvested include machine vision, lasers, and electronic sensors to classify the ripeness of each muskmelon. One machine vision system views the entire width of the melon bed and real-time software identifies the location of fruit which have the size and color of fruit which may be ripe. A second camera is positioned to view individual melons and provide precise guidance information for gathering each fruit. Oscillating fans were used to disturb foliage and expose the melons to each camera. A real-time image coprocessor inside a 486 (25 MHz) Personal Computer found from 82 to 88% of the muskmelons, but with up to 36% false detections. The inclusion of a laser plane projector reduced the number of false detections to 2.1%, but requires additional software and hardware for real-time operation. Studies with a S_nO_2 sensor which measures aromatic volatiles emitted by fruit, revealed a positive correlation between the output and other, independent measures of ripeness. Additional work will be necessary to speed the response of the sensor and integrate it into the robotic grippers. The electronic sensor has much potential apart from robotic harvesting, as a device for quantifying ripeness of climacteric fruit in the field during manual harvesting operations, and in packing houses where fruit are sorted according to ripeness categories.

The prototype selective harvester consists of a 3 axis Cartesian robot designed to meet the requirements of melon harvesting. One axis reaches the full width of the melon bed (1.5 m), a second axis moves 1.5 meters along the row, and a third axis extends and retracts a distance of 1 meter vertically. A clam shell gripper consisting of two rings is attached to the end of the vertical axis by a flexible spring which permits it to conform when objects are encountered unexpectedly. Each axis consists of hardened, precision guide-rails, and carriages which glide on roller-bearing wheels. Electrical servo motors move the carriages via a chain and pulley arrangement. Motion control is effected by a 3-axis motion coprocessor card inside the 486 Personal Computer.

Field testing of the integrated system of sensors, manipulators, servos, and intelligent controls demonstrates the concept of selective harvesting via robotics is valid. Additional research efforts are necessary to develop similar technologies for other crops such as pumpkins, broccoli, cauliflower, cabbage, and extend the utility of the robotic harvester. When used to perform a variety of operations in a multitude of crops, agricultural robotics can be affordable.

ROBOTIC SYSTEMS FOR SELECTIVE HARVESTING: INTEGRATION AND PROTOTYPE TESTS

ACKNOWLEDGMENTS

The successes of this project are due to the combined efforts of many people. From the very inception of this project, Dr. Y. Alper has been a fatherly adviser; he kept us on track, headed in the right direction. Dr. Richard Stroshine and Dr. Kamyar Haghighi at Purdue University guided two research engineers to complete studies related physical properties of melons and the finite element design of the grippers. Both efforts resulted in refereed publications and one effort won a "best paper" award from the American Society of Agricultural Engineers. Dr. Bernard Engel directed Xin Zhuang's study of color image processing using neural networks to determine melon location. Dr. James Simon, Professor of Horticulture, provided the expertise on melon ripeness and aromatic volatiles, and guided much of Meny Benady's studies of the "sniffer". Dr. Simon and Denys Charles contributed significantly to the analysis of volatile compounds emitted from ripening fruit. Dr. Simon Nof (Purdue) provided valuable expertise on industrial robotics and systems analysis of path planning. Dr. Tamar Flash and Vadim Rogozin developed and tested a theory for control of robots used in unstructured, dynamic environments such as agriculture. We are grateful for all their contributions.

The efforts of the prototype construction and evaluation team (Yael Edan, leader, Yuri Dobrusin, Josef Grinsphun, and Itzhak Wolf) are especially commendable. Itzhak Wolf's contributions to the mechanical design of the robotic manipulators and grippers made a field prototype possible. His idea to power the tractor motion by reverse driving the PTO with a hydraulic motor, which was conceived many years ago, was of great value to this project. Few people outside NASA have ever attempted to operate such delicate equipment in such hostile environments: heat, rain, dust, vibrations. Yet, by working long, tiring hours in those combative conditions, and on an extremely tight budget, they succeeded where few ever attempted. Congratulations! You are truly a quality team.

To the management and staff of Kibbutz Yifat and the South Field Station (Moshav Ein Tamar) we owe many thanks for providing melon plots to field test the prototype. Without their contributions, we would have been required to divert critical resources away from constructing and testing the prototype, and success would have been jeopardized. Thanks.

Often overlooked in a project this size are contributions by graduate students. The position description "research associate" truly applies, for they were over-worked and under-compensated:

- Dr. Martha Cardenas-Weber, thanks for developing the initial codes to locate melons;
- Dr. Meny Benady, thanks for developing and evaluating the laser-machine vision system and especially for developing the "sniffer";
- Dr. Amots Hetzroni, thanks for your contributions to the real-time image system; and
- Dr. Yael Edan, thanks for performing the simulation studies which produced the design criteria for the field prototype.

To the men and women of BARD offices in both the US and Israel, we express our sincerest gratitude for the funding which made this all possible. You are a great group to work for.

From the time Yael Edan joined the team, positive things began to occur. In virtually every aspect of this project, she has made very substantial contributions. Her abundant energy and enthusiasm stimulated all of us to strive for excellence. Thank you Yael. May the world come to recognize and respect your professional capabilities and have the privilege of knowing the quality of your personal character, as we do.

To all of you, please accept our sincerest thanks for making this a successful project. Congratulations. May God bless. Shalom!

Gaines E. Miles, Purdue

Uri Peiper, Volcani

ROBOTIC SYSTEMS FOR SELECTIVE HARVESTING: INTEGRATION AND PROTOTYPE TESTS

TABLE OF CONTENTS	PAGE
ABSTRACT	i
ACKNOWLEDGMENT	ii
1. INTRODUCTION AND OBJECTIVES	1
2. RESEARCH ACTIVITIES.....	2
2.1 Sensors	2
2.1.1 Machine Vision for Detecting Melon Location	2
2.1.1.1 Visual Parameters of Melons.....	2
2.1.1.2 Real-time Image Processing Methods.....	2
2.1.1.3 Vision Hardware and Software Design.....	3
2.1.1.4 Camera Calibration	3
2.1.1.5 Gray-Level Based Melon Location Algorithm	4
2.1.1.6 Color-Based Melon Location Algorithm	7
2.1.1.7 Field Experiments with Gray-Level and Color Vision.....	8
2.1.1.8 Results of Field Studies with Gray-Level Imaging to Locate Melons	9
2.1.2 Sensing melon location with lasers and image processing.....	10
2.1.2.1 Development of Basic Technologies for Locating Melons with Structured Light	10
2.1.2.2 Results of Field Experiments with the Laser	14
2.1.3 Detecting Melons with Near Infrared (NIR) Vision	15
2.1.4 Locating Melons with Artificial Neural Networks	16
2.1.5 Melon Ripeness Sensors	18
2.1.5.1 Melon Ripeness Determination by Aromatic Volatiles	18
2.1.5.1.1 Electronic Sensing of Aromatic Volatiles	19
2.1.5.1.2 Electronic Sensor Evaluation Procedures	19
2.1.5.1.3 Results of Electronic Sensor Evaluation	21
2.1.5.2 Melon Ripeness Determination by Image Features	24
2.1.6 Summary of Sensor Developments.....	24
2.2 Intelligent Control of a Field Crops Robotic Machine.....	26
2.2.1 Superposition Scheme for Real-Time Trajectory Modification.....	26
2.2.1.1 Description of the Superposition Scheme	26
2.2.1.2 The Mathematical Model	27
2.2.1.3 Real-Time Implementation of the Superposition Strategy	29
2.2.1.4 Optimum Switch Time in the Case of Velocity Constraints.....	33
2.2.1.5 Optimum Switch Time for Equal Sign Amplitudes	33
2.2.1.6 Optimum Switch Time for Different Sign Amplitudes.....	34
2.2.1.7 Complexity of the Superposition Algorithm	34
2.2.1.8 Implementation of the Superposition Strategy	35
2.2.1.8.1 Adept-One System Overview and Description of the Experiments	35
2.2.1.8.2 Results for the Adept-One Robot.....	37
2.2.1.8.3 Results for the Robotic Melon Harvester.....	40
2.2.2 Animated Simulation and Analysis of Robotic Systems	41

Table of Contents, continued...	Page
2.2.2.1 Animated Simulation and Analysis of Industrial Robotic Tasks	42
2.2.2.2 Simulation of Agricultural Robots	46
2.2.2.3 Simulation of Intelligent Control Systems.....	52
2.2.2.4 Robotic Melon Harvesting Model	53
2.2.2.5 Simulation Results	55
2.2.3 Basic Design Concepts of a Selective Harvester for Muskmelons.....	57
2.2.4 Overall System Design.....	58
2.2.4.1 Motion Controller	59
2.2.4.2 Signal Amplifiers	61
2.2.4.3 Signals Switchboard.....	61
2.2.4.4 Gripper State Sensors	62
2.2.4.5 Path Traveled Sensor	62
2.2.4.6 Carrier Turn Angle Sensor	62
2.2.4.7 Movement Decoder	62
2.2.4.8 Optoisolated, Solid State Relays Block	63
2.2.4.9 Power Supply Blocks.....	63
2.2.4.10 Sensors Block.....	63
2.2.4.11 Emergency Block.....	63
2.2.4.12 Electrical Power Sources	63
2.2.4.13 Cooling Systems	64
2.2.4.14 Tractor Control.....	64
2.2.5 Software Design	64
2.2.6 Software and Hardware Integration.....	68
2.2.6.1 Preliminary Integration.....	68
2.2.6.2 Step-Mode Software.....	68
2.2.6.3 Continuous-Mode Software	70
2.3 Prototype Design	73
2.3.1. Construction of the Cartesian Manipulators.....	73
2.3.1.1 Frame Members	74
2.3.1.2 Slide Systems.....	75
2.3.1.3 Drive Elements	75
2.3.1.4 Position and Track Limit Control	76
2.3.1.5 Protective Elements	77
2.3.1.6 Umbilical Cords for Power and Control	78
2.3.2. Design of the Carriage and Frame.....	78
2.3.2.1 The Frame	78
2.3.2.2 Foliage Fan	80
2.3.2.3 Light Tent.....	81
2.3.2.4 Collecting Troughs for Harvested Melons	82
2.3.2.5 Reflection-Brightness-Level References	82
2.3.2.6 Lighting Fixtures.....	82
2.3.2.7 Gripper Design	82
2.4 Field Tests.....	84
2.4.1 Field Procedures	84

Table of Contents, continued...	Page
2.4.2 Performance results	84
2.4.2.1 Sensing Performance.....	84
2.4.2.2 On-line Mode Testings.....	85
2.4.2.3 Fruit Picking.....	86
2.5 References	88
3. EVALUATION OF RESEARCH ACCOMPLISHMENTS & CONCLUSIONS. ...	93
4. COOPERATION	95
5. PUBLICATIONS	96
5.1 Articles in Refereed Journals.....	96
5.2 Book Chapters.....	96
5.3 Dissertations	96
5.4 Papers and Proceedings Presented at National and International Conferences	97
5.5 Manuscripts Submitted to Refereed Journals.....	97
5.6 Departmental Reports	98
5.7 Citations and Articles in the Popular Press.....	98
6. APPENDICES	99
6.1 Appendix A (Raw Data From Field Tests)	A-1

ROBOTIC SYSTEMS FOR SELECTIVE HARVESTING: INTEGRATION AND PROTOTYPE TESTS

1. INTRODUCTION AND OBJECTIVES

Selective harvesting requires sensors to locate each ripe fruit, grippers to delicately grasp and detach the fruit, and manipulators to translocate the fruit into shipping containers. Before the advent of robotics, these laborious, repetitive tasks were performed by humans. Now it is possible to program machines to perform these stoop-labor tasks. But unlike counterparts in manufacturing, agricultural robots must perform a wide variety of tasks, work in chaotic environments, and be exposed to dust, moisture and temperature extremes.

The value of an agricultural robot not only lies in its ability to perform tasks (such as selectively harvesting only the ripe fruit), but also in its ability to be reprogrammed and adapted to perform other tasks. A versatile machine can be used more hours each year, and thus the cost per hour is less. Even though a robot may be developed for a specific task, if it not only does that job, but others as well, it becomes a much more valuable machine. Thus, even though at times, it may have been easier and less expensive to design fixed automation to perform a specific task, that was avoided whenever possible, in order to provide maximum flexibility and increase the number of tasks which the machine could ultimately perform. Thus, in this project, the robot should not only harvest muskmelons, but harvest crops such as broccoli, cauliflower, lettuce, cabbage, etc. By performing these tasks, the robotic harvester gives the owner the option of growing a mix of crops, or rotating crops each year due to pest management practices, or economic reasons. Fixed automation would require a unique machine for each crop, and in the long-run be much more expensive than a robot.

The basic concepts for sensors and manipulator controls were developed in a previous study (Miles, 1990). The details of those concepts are contained in the final report (Miles, 1990) and in other publications (Cardenas-Weber, *et al.*, 1989; Cardenas-Weber, *et al.*, 1991; Cardenas-Weber, *et al.*, 1991; Edan and Miles, 1989; Edan, *et al.*, 1989; Edan and Miles, 1991; and Wolf, *et al.*, 1990) and will not be repeated, except to identify and document changes necessitated by discoveries during field and lab studies. Without a doubt, the simulation study provided much needed insight into the requirements for a field prototype, and contributed substantially to the success of the field studies. The simulations helped focus research studies onto feasible ideas, and the data collected in the field not only confirmed machine performance, but provide feedback for additional modeling efforts to improve the design of the next version.

The general objective of this work was to integrate the melon gripper, machine vision sensing of fruit location, and robotic motion simulations into a field prototype machine. The specific objectives were:

1. To improve and extend the fundamental technologies of gripping, sensing, and manipulator control;
2. To integrate the individual, intelligent components (the robot controller and the image processor) under a central control processor; and
3. To design, construct and field-test a prototype robotic melon harvester.

2. RESEARCH ACTIVITIES

2.1 Sensors

2.1.1 Machine Vision for Detecting Melon Location

2.1.1.1 Visual Parameters of Melons

Melons (*Cucumis melo*, L.) are delicate, fleshy fruits. They grow randomly scattered over the beds either singly or in small groups and are hidden by leaves. Melons do not ripen at the same time. Thus, every fruit must be detected separately and evaluated for ripeness prior to harvesting. The important melon characteristics (Kasmire, 1981) which can be used for visual detection are:

- **Shape:** Melons are round to elongated; some varieties have meridian ribs.
- **Size:** The ripe melon diameter is variety-dependent and varies from 10 to 15 cm for small varieties to 15 to 30 cm for larger ones.
- **Color:** Melon color changes from dark green at the early stages of growth to creamy and yellow at the mature stages.
- **Texture:** Mature melons have tight, well developed netting which is rounded and well raised above the skin surface.

The detection and evaluation of melons is a complex task, due to the variability of the object features and influences of the unpredictable changing environment. The visible features of natural objects, such as melons are not well defined, since they vary in size, shape, texture and color. Furthermore, they depend on random environmental influences such as illumination, shadows, occlusion, and weather. The dynamic nature of the robotic harvester further complicates the sensing process. The melon locations must be acquired, organized and updated in conjunction with the harvesting and moving operations.

The surface reflectivity of melons is greater than most soil and leaf surfaces (Cardenas-Weber, *et al.*, 1991; Cardenas-Weber, 1991). Image processing algorithms based on brightness alone are able to detect 89% of the melons, but with considerable number of false detections. As a result, this project studied additional methodologies for accurately detecting fruit location.

2.1.1.2. Real-time Image Processing Methods

The following image processing techniques were evaluated for detecting melons:

- grey-level,
- color,
- infrared, and
- structured light.

All experiments were conducted in the continuous-mode of operation, i.e., images were snapped while the tractor was advancing along the melon bed. All analyses were conducted independently, i.e., with no connection to the robot operations. Grey level and color imaging were evaluated for different fields of view and resolutions. The experiments were conducted for two different melon cultivars, in two different seasons at two different locations: Ein Tamar

(Jordan Valley; cv. Galia) and Kibbutz Yifat (Israel Valley; cv. 14E) in the spring (April) and summer (July) 1993 seasons. Experiments were conducted in two different seasons which covered two different types of melons at two different locations.

2.1.1.3. Vision hardware and software design

The vision hardware consists of two cameras and a real-time image processing system. The vision sensor is a NEC NC-15 CCD camera equipped with a FUJINON 6 mm auto-iris lens and option of automatic black/white balance. It provides both color NTSC and RS-170 output with a 683 X 492 resolution. The image processing hardware is a PC-based Matrox Image Series Board Set which includes three boards that are interconnected through an internal bus:

1. The Image 640 (medium resolution) Base Board is the central co-processor. It is responsible for communication between the host (PC) and the other vision boards and also provides display and graphical functions;
2. The Image-CLD is a color multi-channel digitizing board with RGB and PAL options; and
3. The Image-RTP is a real-time processing board. The Image-RTP board has pipelined architecture and hosts Cascaded ALU, Neighborhood and Statistical Processors for up to 15 MHz convolutions, morphological and statistical operations. The hardware is supported and surrounded by the board-resident pipelined-architecture software (Image-Shell) which must be loaded prior to operation.

The vision software was developed under the MS DOS Microsoft C (ver. 6.00) environment. It utilizes the Matrox Programmer's Kit C-libraries and drivers providing C-program access to the vision board functions. A bi-directional oscillating air was constructed to blow the foliage and to expose the melons in the vision area. A plastic canopy covered the whole frame to provide homogeneous illumination of the scene in the natural sun-light conditions. Quartz halogen flood light illumination was used when imaging in the dark. Four black and white reference tiles were mounted at the corners of the camera's field of view to enable on-line calibration of the black and white reference level of the digitizer during image processing.

2.1.1.4. Camera calibration

Camera calibration was conducted for a vertical camera position (Gonzalez and Wintz, 1987). The calibration curves provide data for coordinates (camera-arm) transformation and for calculating the scaling coefficients necessary to transfer original object parameters from its image parameters. The camera height was varied from the maximum height viewing whole melon bed (120 cm width), to the minimum height corresponding to viewing a half melon bed. The calibration was done by placing 31 cm and 62 cm white strips at nine predefined locations on the ground and measuring their correspondent length in pixels. The measurements were repeated for ten different camera elevations.

Variances in light intensity, optics, and geometry of camera setup require calibration. Pairs of black and white reference palettes are positioned at the corners of the field of view, near the level of the fruit. Software has been developed to locate the palettes and to determine the average intensity for the black and white surfaces. From these data, gradients can be determined, which account for differences in illumination of the melon bed. Once the gradient

has been determined, Matrox library functions perform the transformation on subsequent images in real-time.

Because these palettes are permanently attached to the frame, while the camera is not, software was written to determine the pixel locations of their edges. This measures the relative position (height and orientation) between the camera and the vehicle frame, and determines the translation between the robot origin and the camera origin.

In order to determine scale of objects in the image, a white, circular plate is placed in the field of view. Software has been written to locate the plate's center, and to measure the height and width. Using a circular object eliminates the need to align the target with the camera's axes.

2.1.1.5. Gray-level based melon location algorithm

The melon location algorithm (Figure 1) consists of the following steps: images acquisition, preprocessing, integration, segmentation, morphological analysis and improved segmentation, object recognition, and parameters extraction.

- (1) **Images acquisition.** In order to use maximum primary scene information and to minimize the foliage occlusion impact, several consequent images are grabbed while the harvester is in a fixed position (the first version of the prototype harvester is operated in the step mode). Different images are snapped for different blowing directions. Every image is filtered to remove noise.
- (2) **Preprocessing:** A real-time procedure is developed to adjust the intensity level of each input image. This is achieved by managing a look-up-table (LUT) with respect to four reference tiles located at the corners of field of view of the camera. The average gray level of the reference tiles is used to define the linear LUT for real time calibration of black and white reference level of the digitizer.
- (3) **Integration** is achieved by computing the average of the preprocessed images. The integration highlights the bright ripe melon domains, that remain stable under the air blast, as opposed to the fluctuating leaves.
- (4) **Segmentation** of the average image is obtained by binary constant threshold with a "low threshold" value.
- (5) **Morphological binary opening operation** (Haralick *et al.*, 1987) is applied to remove thin and/or small objects (stems and small bright leaves), while retaining objects with relatively bigger radial size in the segmented image. The result is combined with the integral (average) image using a logical AND operator to highlight the initial gray-scale domains which correspond to blobs in the opened image. A "high threshold" value is defined then as (Mean Standard Deviation) calculated from the intensity distribution in these gray-scale domains. Binary constant threshold with the "high threshold" value is applied to improve the segmentation.
- (6) **Spatial characteristics** of objects in the binary image such as: diameter, area, perimeter, parameters of the minimal rectangle which contains the object, are used for the recognition. Recognition is achieved by comparing these characteristics with the melon features. For every object that is classified as a melon, the planar coordinates and estimated diameter are extracted. The outputs of the algorithm are the total number of melons and the parameter list which include X and Y coordinates.

Figure 1. Flow-chart of the gray-scale vision location algorithm

The location algorithm was implemented in two modes: interactive and operational. The following options are available in the interactive mode:

- a) Modify processing parameters to adjust the system for melon variety or acquisition conditions;
- b) Plot the histograms;
- c) Dump images and other information to files;
- d) Display the main processing stages;
- e) The interactive camera calibration procedure is done by combining the imaging and the graphical vision board capabilities. This provides a live image of the camera calibration scene simultaneously with a graphic tool to extract the objects' coordinates from the image.

The operational mode includes only the algorithm's body. Most of the location algorithm is implemented using the vision system hardware functions. This speeds up the analysis and saves the host computer processing time for other robot control activity. Functions that are not available on board, are executed on the host computer. Among these are general control, some fragments of the histogram analysis, object recognition and parameters extraction steps. The hardware implementation of the image processing speeds up the analysis and saves the host computer processing time for other robot control activity. The constant threshold value for the segmentation is set before the processing. The threshold value can be changed on-line to adapt to different lighting conditions using the interactive mode. Two morphological opening techniques (composition of erosion and dilation): distance-based and Min-Max based, were tested to obtain desired morphological filtering results. The fastest response was obtained using the distance operator; it takes four passes (each up to 15 MHz) to produce an image opening. The distance function determines the shortest distance between each object pixel and its object's edges and can subsequently perform a fast erode or dilate operation on the image. Another way to achieve opening of a binary image is to use the Minimum operator for erosion and the Maximum one for dilation. These operators replace each pixel by the result of Min or Max operation in the pixel's neighborhood. This results with less distortion to the objects' shape and requires one pass per convolution. The number of convolutions depends on the desirable degree of erosion or dilation. Since both parameters,

object shape and execution time are important, the two morphological filtering approaches are combined: the distance based erosion following with the Max-based dilation provide the opening effect.

Objects in the binary frame are labeled to allow individual manipulation. The recognition process consists of two steps: preliminary evaluation step and recognition. Minimal and maximal melon diameters are set corresponding to the melon variety. The binary blobs areas and diameters are evaluated in the preliminary step. Blobs which are too small, as well as those which are too big are discarded. The recognition module must be adapted to the melon variety. Two parameters were proved to be useful for recognizing the round melon like shapes in the images:

$$PAR_1 = \frac{PERIMETER^2}{AREA} \dots\dots\dots [1]$$

$$PAR_2 = \frac{AREA}{DIAMETER^2} \dots\dots\dots [2]$$

where: AREA and PERIMETER are counts of pixels occupied by the object and it's boundary; DIAMETER is the largest of the X and Y object's dimensions. These parameters evaluate roundness of an object shape. Parameter PAR₁ additionally evaluates the smoothness of the objects' boundaries, which helps avoid false detection of jagged leaves-stems combinations with a big area. The following relations are used to characterize the melon-like shapes in the images:

$$PAR_1 < VAL_1 \dots\dots\dots [3]$$

$$PAR_2 > VAL_2 \dots\dots\dots [4]$$

Values VAL₁ and VAL₂ must be determined statistically for every melon variety. It can be done by evaluating averages AVG₁, AVG₂ and standard deviations STD₁, STD₂ of melon parameters PAR₁, PAR₂ distribution during a preliminary processing of a set of control images. Values VAL₁, VAL₂ were derived as follows to provide approximately 95% correct melon detection in the control set of images:

$$VAL_1 = AVG_1 + 2 * STD_1 \dots\dots\dots [5]$$

$$VAL_2 = AVG_2 + 2 * STD_2 \dots\dots\dots [6]$$

More fruit detection can be provided by increasing VAL₁ (decreasing VAL₂), but this will increase the false detection probability. For example, if VAL₁ = AVG₁ + 3*STD₁; VAL₂ = AVG₂ - 3*STD₂, then 98% of fruit can be detected in the control images, but with 14 cases of false detection of leaves.

Objects in the binary frame are labeled to allow individual manipulation. The process of recognition consists of two steps: preliminary evaluation step and recognition. Minimal and maximal melon diameters are set corresponding to the melon variety. The binary blobs areas and

diameters are evaluated in the preliminary step. Blobs which are too small, as well as those which are too big are discarded. In the recognition pass rules (3),(4) are applied for melon discrimination. The average execution time per algorithm cycle was estimated experimentally as $(1 + 0.1 N)$ sec, for the HP Vectra (25-MHz 80486 processor) PC host computer, where N is the number of separate live input images. The images in Figure 2 illustrate the major steps of image analysis: integration, segmentation, morphological opening and recognition.

2.1.1.6. Color-based melon location algorithm

For color images, the robot far-vision consisted of a "Panasonic" PAL video camera with a 720x576 resolution. The PAL video signal is translated to the RGB-standard to split the composite signal into three color components,- the Red, Green, and Blue on the Matrox Image CLD digitizing board. The RGB-image components were stored in three different frame buffers to provide possibility of color analysis.

Color difference between ripe fruits and their leaves exist for many melon varieties. This can be used as a basis for melon location. The full color of a ripe yellow melon surface in most cases is composed of "more red than green" as opposed to a green-yellow leaf surface, which is "more green than red".

Figure 2. Integration, segmentation, morphological opening and recognition performed on the same image.

The location algorithm consists of the following steps (Figure 3):

- (1) Image acquisition provides a full-color RGB-image.
- (2) Preprocessing. The Red and Green image components are filtered to remove noise.
- (3) Binary image is a result of the Red component constant threshold.
- (4) Morphological opening is obtained by erosion and dilation of the binary image.
- (5) Objects in the opened binary frame are labeled to allow individual manipulation.
- (6) The process of recognition consists of two steps: preliminary evaluation step and recognition. The binary blobs areas and diameters are evaluated in the preliminary step. Both small and big blobs are discarded. To recognize a ripe (yellow) melon, the following relation is applied then for each blob:

$RED > GREEN$ [7]

where Red and Green are average intensities of the blob's center 5X5 neighborhood in the Red and Green components of the RGB-image. Parameters (planar X, Y coordinates and estimated diameter) are extracted for each object which was identified as a melon during the recognition.

The color based location algorithm was implemented in two modes: interactive and operational similar to the gray-level based algorithm implementation. The algorithm's average cycle execution time was estimated experimentally as 1.3 sec, for the HP Vectra (25-MHz 80486 processor) PC host computer.

Figure 3. Flow-chart of the color vision location algorithm

2.1.1.7 Field experiments with gray-level and color vision

During the gray-level based imaging experiments, series of images were snapped and dumped to files using the robot vision system (far-camera) and specially designed software. In the color based imaging experiments, the scenes were recorded using a "Panasonic" PAL video camera that observed the same area as the robot far-camera. Later, during laboratory analysis, the gray-level images were loaded to the system; the color video images were captured directly from a VCR. The included analysis of cultivar Galia images obtained from an irrigated melon field. A row 60 cm wide was used for the gray-level and color images acquisition under the

robot blower activity. A total number of 55 ripe melons with 8-13 cm diameters were registered visually in the gray level images and in the video-record of the same area. The melon bed was covered with clear polyethylene mulch. The summer experiments included analysis of images obtained from an non-irrigated melon field without using the blower. A total number of 210 ripe melons with diameters 8-13 cm were registered visually in the gray-level images. Three hundred and fifty ripe melons were registered in the video-record of an other field area. While the images acquisition all the system moved along the melon bed at approximately 5 cm per second. The summer experiments included the following:

- a) "full-row width" gray-level images acquisition (natural sunlight conditions);
- b) "half-row width" gray-level images acquisition (natural sunlight conditions);
- c) "full-row width" color images acquisition (natural sunlight conditions);
- d) "full-row width" color images acquisition (artificial lighting conditions in night);

2.1.1.8 Results of field studies with gray-level imaging to locate melons.

For field experiments conducted during the Spring of 1993, 82% of the melons were detected by the gray-level real-time vision using the two-input-images mode, and 83% were detected using the four-input-images mode. Non-detection cases were results of the foliage occlusion (12%) and small melons, - less then 10 cm in diameter (6%). 12 cases of false detection were registered: 5 cases were represented by big bright leaves which reached 13 cm in their diameters and 7 cases were represented by bright patterns of the plastic mulch.

The color based algorithm was able to detect 82% of melons. Non-detection cases were results of the foliage occlusion (14%) and small melons, - less then 10 cm in diameter (4%). 15 cases of false detection were registered: 7 cases were represented by big yellow leaves and 8 cases were represented by patterns of the plastic mulch.

The following results were obtained during the Summer:

- (a) 87% of the melons were successfully detected, most (10%) of the non-detection cases were due to occlusion, 3% of non-detected melons were formed by small fruits (less then 10 cm in diameter). 22 cases of false detection of big bright leaves were registered.
- (b) 85% of the melons were successfully detected. 7% of the non-detection cases were due to occlusion; 5% - were not detected because their images included a small part of fruit which "was cut" by the image frame boundary. 3% of non-detected melons were formed by small fruits (less then 10 cm in diameter). 24 cases of false detection of big bright leaves were registered.
- (c) 86% of the melons were successfully detected in full-color images using color based image analysis. 9% of the non-detection cases were due to occlusion, 5% of non-detected melons were formed by small fruits (less then 10 cm in diameter). 36 cases of false detection of big yellow leaves were registered.
- (d) 88% of the melons were successfully detected in night experiments under the artificial light conditions. (9%) of the non-detection cases were due to the occlusion effects, 3% of non-detected melons were formed by small fruits, 26 cases of false detection of big bright leaves were registered.

The results of field experiments are summarized in Table 1.

Table 1. Real-Time Vision Field Experiments Results for Spring and Summer, 1993.

Experiment	Spring	Spring	Summer	Summer	Summer	Summer
Method	Gray-Level	Color	Gray-Level	Gray-Level	Color	Color
Number of Melons	55	55	210	210	350	350
Diameter (cm)	8 - 13	8 - 13	10 - 20	10 - 20	10 - 20	10 - 20
Field of View	Full Bed	Full Bed	Full Bed	Half Bed	Full Bed	Full Bed
Illumination	Sunlight	Sunlight	Sunlight	Sunlight	Sunlight	Artificial
% Detected	82	82	87	85	86	88
% Not Detected	18	18	13	15	14	12
Number (%) False Detections	12 (22%)	15 (27%)	22 (10%)	24 (11%)	36 (10%)	26 (7%)

The results show approximately equivalent performance of the gray-level and color-based algorithms for the tested cultivars. It is clear that the fruit dimension (or more exactly the ratio of the fruit and it's leaves diameters) is the critical parameter: the bigger fruits the better the detection results. The image resolution (the camera height or zoom) must to be optimized for each fruit variety. A bigger resolution is necessary to detect small fruits, however, if the resolution is too big, the recognition is reduced since more fruit "are cut" by the image frame boundary. This was the reason that the system performance was better in experiments with the full bed view than in the case of a half bed view. Hence, optimal performance must be determined experimentally.

Decreasing the region of interest decreases both the image processing time and frame buffer memory requirements. The region of interest dimensions can be decreased by sub-sampling the source image by some sampling factor number FACT. In order to evaluate the effect of sub-sampling, only every FACT rows and columns of source data are written back to the frame buffer when the sub-sampling is applied. FACT=1 corresponds to processing with the original resolution of the region of interest. The performance of the image processing algorithms with sub-sampling was tested by using four sets of images (with 100 images in each set), collected during the Summer 1993 experiments a),b),c),d). The results are summarized in Table 2. The results indicate that sub-sampling with a sampling factor 2 can be effective/sufficient under appropriate time and/or memory requirements for the image processing.

Table 2. Image processing results for different resolutions

Sampling Factor	Gray-Level Vision			Color Vision		
	Sunlight	Sunlight	Execution Time (sec/image)	Sunlight	Artificial Light	Execution Time (sec/image)
	Full Bed	Half Bed		Full Bed	Full Bed	
	% Detected	% Detected		% Detected	% Detected	
1	85	84	1.10	86	87	1.31
2	82	80	0.71	83	84	0.82
4	75	72	0.51	78	80	0.63
8	62	60	0.48	65	69	0.59

2.1.2. Sensing melon location with lasers and image processing

2.1.2.1. Development of basic technologies for locating melons with structured light

Methods of detecting melons using structural light were developed at Purdue (Benady and Miles, 1992; Benady, 1993). Light striping is a particular case of the structured light technique for range sensing. A light striping range sensor consists of a line projector and a camera that is displaced from the light source; the line projector casts a sheet of light onto the scene. The projection of the light plane on the surface results in a line of light that is sensed by the camera (Figure 4). An elevation profile of the scene can be obtained by transforming the detected light points from image coordinates to world coordinates via calibration.

There is a trade-off in the selection of α - the camera tilt angle. By selecting small values of α an illuminated point will be detected for every Y value. But, as α approaches 0, the vertical resolution is reduced, to such an extreme that when the camera is parallel to the light source, there is no occlusion, but also no height information is available. Conversely, as the vertical resolution is increased by increasing α , up to a maximum of 90° , the probability of occlusion and loss of information increases.

The classical Hough technique for curve detection is applicable if little is known about the location of a curve in an image, but its shape can be described analytically (e.g., a straight line, a circle or an ellipse). The Hough transform approach applies a coordinate transformation to a picture such that all the points belonging to a curve of a given type map into a single location in the transformed space. Its main advantages are that it is relatively unaffected by gaps and noise in the image. The approach was originally proposed by Hough, (1962) for identifying collinear points in digital images. Duda and Hart, (1972) modified Hough's original transform by proposing the use of normal parameterization, and by dividing the parameter space into a grid which are called accumulator cells. By specifying an acceptable error in the parameter values, the computational overhead can be reduced. The circular Hough Transform (CHT) is a variation of the Hough transform for locating circles, or part of circles, in a digital image (Ballard and Brown, 1982). The Hough method organizes the image points into circles by considering all possible circles at once and rating each one on how well it explains the data.

The structured light source used was a PLL 301 laser line projector (Melles Griot Corporation, Irvine, CA). This is a low power (3--4 mW) laser which uses a cylindrical lens to emit a fan of light, in a field angle of 18° with 1 mm beam width at exit aperture. The wavelength of the emitted light is 674.7 nm (visible red). A NEC NC-15 CCD camera, which has a 683 x 492 pixel array with a Fujinon CCTV automatic iris 6 mm lens was used in the development effort. The horizontal field angle of the lens, ϕ_h , was 56° and the vertical field angle ϕ_v , was 43° . The camera was placed at an angle of 46° with respect to the light plane, at a horizontal distance $L=90$ cm, 86 cm above the ground. All measurements were relative to the center of the attachment ring of the lens to the camera body.

Once the illuminated points in the image have been detected and transformed to world coordinates via the calibration procedure, the range data acquired must be interpreted and melons located. The next step of the process is to identify which data points of the scene cross-section are part of a melon, and which are not. The approach for location of melons is based on a modification of the circular Hough transform (Benady, 1993). As opposed to the traditional

circular Hough transform which regards the quantitative support for presence of a circle in image, *i.e.*, the number of data points supporting it, the modification regards also the distribution of data points along its perimeter.

Figure 4. The optical arrangement of the structured light profiling system. The sheet of light projected onto the melon and ground appear as a curve in the detected image.

The circular Hough transform was implemented efficiently by using both domain knowledge to restrict the number of data points tested by the algorithm, and by using domain specific rules to restrict the search space and to validate or refute the outcome of the CHT algorithm. A four-valued labeling scheme was introduced for interpretation of cross sections of the calibrated range map. The four labels, unknown, melon, plant and ground are used to mark the individual data points. Initially all points are labeled as unknown. As the algorithm proceeds data points are labeled and the number of unknown data points that it must treat decreases. The labels are modified by a set of rules until a consistent interpretation of the scene exists. This information leads to location and sizing of melons in the image.

Most of the rules used are very simple and do not require extensive computing: Considering the range of expected melon radii and the ground-error, certain data points may be classified as ground or as leaf if they are too close to the ground or too high above. The perceived melon cannot float in the air or be sunken in the ground, the melon must be close to or resting on the ground, within the ground-error margin. The center of a melon is always below its peripheral points (since laser projection is from above). A melon cannot contain data points within its volume. This rule can either disqualify the melon or discard the interior data points as noise based on the *a priori* probability γ . Two melons cannot occupy the same space simultaneously, disqualifying the candidate that is less probable. The maximum number of melons appearing in a cross-section is determined by the melon size and row width.

The implementation of the Hough transform and of the above observations require determination of task-specific parameters: The maximal ratio between the longitudinal radius and the equatorial radius, previously defined as R_l/R_e . The maximum and minimum radius of

fruit expected. This refers to R_e - the radius at the equator of the fruit. To guarantee that large circles do not prevail in the selection of the highest peak in the accumulator array of the CHT, due to their greater circumference which may include more points, the accumulator values are divided by a normalizing factor, β . A ground variation of ± 30 mm was assumed. The Hough algorithm was implemented with a resolution of 10 mm in the parameter space for a , b and r . The normalization factor, β , was selected manually to be $\sqrt{r}$. The perimeter of the circle was divided into segments of 12° . A necessary (but not sufficient) criteria for considering a collection of points on a radius as consisting a circle was that at least five angular segments be inhabited. The segments were not required to be continuous, to allow for occlusion by leaves. The probability of an interior point inside the melon due to noise is defined by the parameter γ in the algorithm. The value of γ depends on the optical setup and lighting. Under good conditions such as the darkened area used here or by use of an optical bandpass filter $\gamma \rightarrow 0$. The current program assumed $\gamma=0.3$. This allowed for up to 30% interior light points for images acquired under illuminated conditions. Images of darkened scenes were not affected by this parameter and could have also used a value of $\gamma=0$ without any effect on the output.

In one image, there are two full-sized and one small melon on the right side. However, only a single one, off the center of the image to the right, intersects with the light plane. The laser light cannot be seen clearly in the image due to the ambient lighting. Columnwise peak detection of a darkened field image of the identical scene resulted in only the projection of the laser line on the exposed subsection of the melon being detected. The only exceptions are a couple of intensity peaks in the left side of the image that may be due to reflection of the laser light off a leaf. The intensity of light reflected from leaves and soil is significantly weaker than that reflected by melons. This resulted in many occurrences of missing points, but did not hinder the detection of fruit. On the contrary, it serves as a natural filter for extraction of melon data and it also increases the efficiency of the search since fewer data points need to be classified. In many cases, only the fruit reflected the laser line at such a level that it exceeded the detection threshold.

The melon location algorithm labeled each range data point detected in the scene as melon, leaf or ground. Figure 5 shows the interpretation of the scene derived. The nominal ground level is shown by a line, and the ground error tolerance ± 30 mm by the dotted lines. The classified data points are shown in different gray levels. The data points identified as melon are in black, leaf are in light gray and ground in dark gray. The algorithm detected two melons (gray circles) which are labeled 1 and 2 in order of certainty, *i.e.*, the melon labeled 1 was identified at a higher degree of certainty than the melon labeled 2, though they both met the location criteria. Their estimated size is 75 mm (radius). One melon is entirely in the area covered by the laser light while the other is only partly covered.

While the input image size is 640×246 , (157,440 pixels) a single scan of the image reduces the problem size to no more than 640 data points (one per column of the image array) many of which may be immediately labeled and discarded by the absolute-height rule. In practice, the original problem size was nearly always under 350 data points prior to any analysis and was typically reduced to less than 160 points. If only darkened scene images are considered, the numbers decrease significantly to about a quarter of the above, *i.e.*, to less than 90 data points which were typically reduced to about 40 following the absolute-height labeling procedure. The decreased problem size, together with domain specific knowledge that restricts

the search space of the Hough transform, presents an algorithm which is more adaptive to real-time implementation than a straightforward implementation of the Hough method.

Figure 5. Interpreted view of melon bed with projected laser light

The run-time of the program depends on the number of data points in a cross-section and on their elevation range. The implementation was not specifically oriented for real-time performance. Typical program run-times on the Silicon Graphics platform, were 0.6 to 1 sec. when fruit were present, and 0.1 to 0.2 sec. without presence of fruit. Artificially contrived cases, representing best and worst case situations, ranged from 0.1 to 1.7 sec. of computation time. Assuming a travel speed of 20 cm/sec. over the rows, and a melon diameter of 15 cm, the scanner will see the melon for a period of 0.75 sec. The system could therefore, acquire between 4 to 7 cross-sections while traveling over a fruit unless an exposed part of a fruit is located. The duration of the location algorithm for a positive identification will preclude analyzing more than a single view of a fruit.

2.1.2.2. Results of field experiments with the laser

Field experiments with the laser system were conducted autonomously (without the robot arm) in Yifat (Israel Valley; cultivar 14E) in the summer (July) 1993 season. The goal was to evaluate performance of the melon location algorithm. Night time was chosen to provide optimal acquisition conditions for testing the system. The far-camera of the robot was replaced by a "Panasonic" PAL video camera. The laser system was calibrated. Images of the laser line projections on the melon bed were recorded by the video-camera while the harvester moved along the bed at about 5 cm/sec (the blower not active). The images were analyzed later in the laboratory.

The average execution time per algorithm cycle was estimated experimentally as 5.5 sec for the HP Vectra (25-MHz 80486 processor) PC host computer. A total number of 350 melon profiles were registered visually on a video record of the tested melon field area. The total

number of 185 melon profiles were actually tested during the image processing: some frames were skipped because of the long execution time of the algorithm. The laser system was able to detect 74% of the melons in the scenes tested. All non-detection cases were due to occlusion. Only 4 cases ($4 \div 185 = 2.2\%$) of false detection caused by foliage were registered.

2.1.3. Detecting Melons with Near Infrared (NIR) Vision

Analysis of the image obtained in the infrared range may serve as another way to detect melons in the field. It is a natural assumption that the difference in the heat capacities of the melons and the surrounding objects must cause different rates of heating and cooling and, consequently, temperature differentials which could be recorded by image analysis. A preliminary investigation was carried out which consisted of the following stages:

1. Assessment of the basic possibility of distinguishing a melon against the background of the surrounding objects; and
2. Temperature measurement of the melon and the surrounding objects as a function of the time of day.

At the first stage, video recording of the melon field was done in the infrared range of 7 to 10 microns in the sunlight and after sunset using an "Infometric 610" system (Figure 6). At the second stage, temperatures of melons and open ground patches were measured as a function of the time of day. These measurements were done by placing thermocouples on melon surfaces and on the superficial layer of the ground.

Figure 6. Infometric 610 system

By analyzing the images obtained using the "Infometric 610", it was found that temperature differentials of various objects in the field melons, leaves, and the ground were considerably dependent on the time of day and reached maximum values after sunset. Besides, in contrast to images obtained in the visible spectrum - where leaves are the main source of errors in the location of the melon, it is not particularly difficult to distinguish a melon from the

leaves in the infrared spectrum; this can be achieved by setting a threshold value (Figure 7). The source of errors in this case is represented by open ground patches the temperature of which is commensurable with that of the melons. These measurements were conducted by placing thermocouples on melon surfaces and on the upper layer of the ground. Results indicated that maximum temperature differences were recorded during the time interval between 5 PM and 6 PM. The temperature difference between melons and open ground patches reached 10-11 °C. At the maximum point, the temperature of green melons was 5 °C higher than the temperature of the yellow melons. Additional research is needed to evaluate the potential of this technique.

Figure 7. Infrared Image

2.1.4. Locating Melons with Artificial Neural Networks

Artificial neural networks (Zhuang *et al.*, 1992) have been used in biological systems for classifying based on parameters (Miller, 1992; Rigney and Kranzler, 1989), for quality sorting (Brandon *et al.* 1991; Brons *et al.*, 1992; Whittaker *et al.*, 1991), for prediction (Bolte, 1989; Zhuang and Engel, 1990a), and for image processing (Engel and Zhuang, 1991; Thai, 1991; Zhuang and Engel, 1990b). An explicit advantage is that neural networks solve problems by applying knowledge gained through learning and without assumptions about distribution of data. In order to locate melons based on color images, part of an image can be used to develop a neural network which can then be used to classify the rest of the image and other similar images. The primary objective of this exploratory effort was to develop and test an artificial neural network's ability to locate melons based on classification of pixels in the red, green, and blue color bands.

To accomplish this objective, color video images of muskmelons were collected with a camcorder on July 7, 1991 at Purdue's O'Neal Agricultural Research Center. The images included a variety of locations and appearance of muskmelons. The hardware used in this

project consisted of a VHS video camcorder and a Sun Sparc workstation with a VideoPix board and *vfctool* software.

Conventional image processing was conducted through an image processing tool, *SunIP*, a subset of *SunVision*. The image processing operations included histogramming, thresholding, filtering, and morphological and arithmetic operations. A 3x3 median filter was applied twice to the original image to blur the scattered high-frequency noise. A threshold was applied at a gray scale value of 225, which left some scattered clusters as well as melons. A four-step erosion/dilation was performed to the thresholded image to remove the clusters.

The neural network was configured as a three-layer back-propagation network, including input, hidden and output layers. Adjacent layers were fully interconnected. The input layer was composed of a 3x8 array of binary-coded units corresponding to the red, green, and blue bands of the color images. The neural networks simulator was NASA NETS (Baffes, 1991), which runs on a variety of machines including workstations, PCs, and Macintosh computers. The simulator provides a flexible system for manipulating a variety of configurations of neural networks and uses the learning algorithm of the generalized delta back-propagation (Rumelhart *et al.*, 1986). The NETS software was run on a Sun Sparc workstation.

Melons were located by training the neural network to classify pixels in the digitized images. The network was trained by selecting areas of known categories from the image. Once trained, all the image data were propagated through the trained network. The training was an interactive process. NETS provides two error parameters to monitor and adjust during training: Maximum (Max) and Root Mean Square (RMS). A Max error was the greatest distance among the differences between each actual output and desired output in the output layer, whereas a RMS error was based on the root mean square of the distance errors. The training stopped when the Max error became less than or equal to 0.1. After the training converged, the network was propagated through the remaining image data.

In this preliminary study, the neural network was trained with the sampled data from one image, and used to classify two additional images. The neural network, accurately located all muskmelons in three images. In one image, almost the whole muskmelon is seen by the camera, whereas only parts of melons were detected in the second and third images because of obstructions from leaves. The neural network even found both muskmelons in additional images. This success indicates that muskmelons can be located with neural networks no matter how much of a melon is detected, and how many melons are visible. (Obviously, the melons must be at least partly visible.) In addition, the muskmelon on the upper edge of Image2 was greenish, and the other muskmelons in all three images were orange and yellow. The color difference of this muskmelon did not affect detection by the neural network. Because the neural network looks at each pixel, it misclassified some pixels of leaves and stems as muskmelons. This type of misclassification does not cause problems; the criteria for detection could specify a minimum size of object, or the pixels could be removed by a traditional image processing routine such as filtering, or erosion/dilation.

Based on a limited study with three images, it is concluded that neural networks can be used to locate a high percentage of the melons visible in color images. Because hardware capable of propagating neural networks in real-time is now available, future research should be conducted to integrate this sensing technology with the robotic motion controller.

2.1.5. Melon Ripeness Sensors

A selective harvester must be able to determine the ripeness of fruit before harvest without degrading its quality. Many analytical techniques have been developed for determining ripeness of fruit (Dull, 1986), but most of the nondestructive methods are subjective, based upon visual observation of size, shape, color, changes in the physical exterior such as the development of a net or appearance of the abscission zone or "slip" in cantaloupes (Cantwell, 1990; Kasmire, 1977; Kasmire, 1981). Other techniques are destructive: total soluble solids (tss), ratio of sugars to acids, taste and fruit firmness, and internal color changes and internal ethylene content (Kader *et al.*, 1985; Reid *et al.*, 1970). Recent studies on fruit ripeness have examined a wider variety of methods such as NIR (Dull, *et al.*, 1990), NMR (Cho and Krutz, 1989; Cho, *et al.*, 1991), external fruit firmness (Aubert, *et al.*, 1977), and ultrasonics (Upchurch, *et al.*, 1987; Mizrach, *et al.*, 1989; Farabee and Stone, 1991) aimed at identifying specific stages of ripeness or quality in fruit. Some of these methods appear quite sensitive, but are either destructive in nature, require very expensive and sensitive instrumentation, or are unsuitable for use under agricultural field conditions. None of the more sophisticated techniques have been successfully commercialized, yet.

2.1.5.1 Melon Ripeness Determination by Aromatic Volatiles

During melon ripening both ethylene and aromatic volatiles are emitted. Ethylene, a natural hormone associated with the ripening process, has been extensively studied (Abeles, 1973; Morrison, Baird and Webster, 1979; Reid, 1985). Classified as climacteric fruit because of their behavior during ripening, cantaloupes undergo increased rates of respiration and ethylene production during fruit maturation. Most research on ripening and ethylene has been focused on prolonging storage and shelf-life and has concentrated on the response of harvested melon fruit to exogenous applications of ethylene, in part by measuring internal ethylene and a range of other physiological changes associated with ripening, such as respiration (Reid, 1985). Ethylene diffuses from the interior to outside the fruit (Altman and Corey, 1986; Kendall and Ng, 1988) and can be rapidly measured quantitatively using gas chromatography. Cantwell (1990) and others (Bliss and Pratt, 1979) reported the close association between internal ethylene production and ripening. Pratt (1971) reviewed the role of ethylene and abscission, and suggested that the abscission of the fruit from the vine in melons (known as the development of the slip) occurs during increased rates of endogenous ethylene production. Rates of ethylene production and internal ethylene concentrations in cantaloupe appear to be under genetic control (Kendall and Ng, 1988). The formation of the abscission region can vary by cultivar (Reid, 1985). The development of the slip is an important index of fruit maturity and the stem detachment force is strongly correlated with ripeness. Unfortunately, this maturity index is destructive and related only to an individual fruit and not to a larger population.

Aromatic volatiles also accumulate during fruit ripening for both climacteric and non-climacteric fruits (Kitamura *et al.*, 1976). More than sixty aromatic volatiles have been identified in cantaloupes and significant changes in the composition and relative concentration of aroma compounds occurs during fruit ripening (Horvat and Senter, 1987; Kemp, *et al.*, 1972; Kemp *et al.*, 1973; Yabumoto *et al.*, 1977; Yabumoto *et al.*, 1978). Miyazaki and Ookubo (1989) reported that aromatic volatiles and ethylene were much better indicators of fruit maturity than visual appearance.

2.1.5.1.1 Electronic Sensing of Aromatic Volatiles

Semiconductor gas sensors, mainly S_nO_2 , change conductivity as the concentration of gases such as hydrogen, carbon monoxide, methane, propane, or organic solvent vapors belonging to the alcohol, ketone, ester and benzol families (Figaro, 1990a,b). The conductivity of the sensor also increases with ambient temperature and humidity. The range of sensitivity for the TGS 882 sensor is from 50 to 5000 ppm.

The objective of this research effort was to quantify the dynamic response of an electronic sensor to concentrations of naturally emitted gases including ethylene and aromatic volatiles produced by ripening fruit. Such a system could be used for ripeness determination in selective harvesting, or to sort fruit based on ripeness during post-harvest processing.

The ripeness instrument consists of the sensing head, signal conditioning electronics, and data acquisition hardware. The sensing element is a TGS 882 semiconductor gas sensor, located within a small, flexible, rubber cup which is placed on the fruit surface (Figure 8). Aromatic and non-aromatic volatiles produced by ripening fruit accumulate rapidly within the chamber formed by the cup on the surface of the fruit, and cause the conductance of the sensor to change. The sensor is connected in a simple resistor-divider network, and the analog output signal is digitized and stored in the computer.

Figure 8. Field testing of the electronic sensor capturing the headspace gases naturally emitted by the ripening muskmelon

2.1.5.1.2 Electronic Sensor Evaluation Procedures

Fields of melons were grown at two locations in Indiana - the Purdue University O'Neill Vegetable Research Center, Lafayette, IN, 15 km south of Purdue University, and at the SouthWest Purdue Agricultural Center, located 250 km southwest of Purdue in Vincennes, IN. The cultivars used included 'Mission', a popular Western shipper; 'Superstar' the leading Midwestern or Eastern melon; 'Starship' an eastern melon, and the Israeli developed green-

fleshed cantaloupe 'Makdimon', now renamed 'Mediterranean Delight'. All data were recorded in July and August, 1991. The relationship of the emission of fruit volatiles (aromatic and non-aromatic compounds) and fruit maturity, were examined by sampling and evaluating over 200 fruit of the above cultivars. The following observations were recorded for each fruit:

Visual Classification: Fruit of each cultivar were harvested and classified into one of four ripeness stages:

- **unripe**-late green stage prior to slip;
- **half-ripe** as associated with half-slip stage;
- **full ripe** at the full slip stage; and
- **over-ripe** as associated with post-full slip with external fruit color changes and cracking along stem detachment section, yet prior to severe external physical breakdown.

External Properties: The fresh weight and circumference around the equatorial line of the fruit were measured. External color of the fruit was registered visually on a coarse range of 1 to 5, from dark green to orange.

Electronic Sampling: Each fruit was sampled for 30 sec using the electronic sensor (Figure 24) both in the field, prior to detachment from the vine, and in the laboratory, following a 24 hour period of acclimation. The sensor response was measured on four locations on each fruit, adjacent to the stem end, adjacent to the blossom end, along the equatorial line on the side exposed to the sun, and on the side of the fruit, with the means presented here. In the laboratory testing, the ground spot was used as a reference for the orientation of the fruit

Headspace Sampling: After reaching equilibrium in the laboratory, the fruit were placed in sealed glass containers for a period of 1 hour. Headspace samples of the gas were obtained by withdrawing 2 cc of gas using a syringe and needle. The gas was then directly injected into a Varian 3700 gas chromatograph (GC) for determination of aromatic volatiles, and a Shimadzu GC for ethylene determination as previously described (Simon *et al.*, 1983; Charles and Simon, 1990). Standards were used to identify ethylene and compound identification of all aromatic volatiles were verified by Mass Spectrometry (MS) with a Finnigan 4000 GC/MS using electron impact analysis.

Cavity Sampling: Using a long (5 cm) stainless-steel needle, 1 cc gas samples were withdrawn from the seed cavity of each fruit to determine internal levels of ethylene and aromatic volatiles via GC analysis.

Stem Detachment Force: Stem detachment force was measured by a Chatillon DFGHS strain gauge (Chatillon Inc., Greensboro, NC). A cotton cord was attached to the stem at the location of attachment of the fruit stem to the runner and to the strain gauge on the other end. The stem was then pulled along the direction of growth until it detached, and the peak force directly recorded.

Flesh Firmness: Internal flesh firmness was measured by cutting a 2.5 cm slice of the fruit along the equatorial line, and using a 1.1 cm diameter cylindrical plunger (ASAE standard S368.1) attached to an Ametek CADET (Ametek Inc., Hatfield, PA) strain gauge mounted vertically to a manual drill press. The value registered for each fruit was the average of three readings sampled at three locations, equidistant from the external rind and the seed cavity.

Total Soluble Solids (TSS): TSS were measured from the same slice used for the flesh firmness measurements. Flesh samples were taken from three locations, and were manually

crushed, then sieved through sharkskin paper, and the percentage of soluble solids was directly read through an ATAGO ATC-1 hand-held temperature compensated refractometer (Atago, Inc. Japan).

2.1.5.1.3 Results of Electronic Sensor Evaluation

Nine aromatic volatile compounds were detected in the headspace gases of the ripening melon fruit (Figure 9). There was a strong increase in the number of compounds emitted from the muskmelon cv. Mission as the fruit ripened, and an increased emission rate of CO₂ and aromatic volatiles. The extremely low level of aromatic volatiles emitted by the unripe fruit is particularly significant. The levels of CO₂ produced by the fruit are shown adjacent to the emissions of the aromatic volatiles. Our results of these volatile gases captured by headspace sampling, confirm previous reports by others (Horvat and Senter, 1987; Kemp *et al.*, 1973; Yabumoto *et al.*, 1978), that the ripeness stage of muskmelons is related to the production of aromatic volatiles.

Figure 9. Aromatic volatile constituents emitted from ripening muskmelon, cv. *Mission*
(Average of 4 fruit per ripeness stage)

Figure 10 illustrates typical dynamic responses of the electronic sensor. After 5 seconds the readings among the fruit ripeness classes are clearly distinct. Capturing the headspace volatile gases surrounding the melon fruit, our electrical sensing system emits an increasing signal correlated with both quantitative and subjective indices of fruit ripeness (Figure 10) including external fruit color, stem detachment force, flesh firmness, total soluble solids, ethylene and aromatic volatiles (Table 3). The electrical sensor appears to primarily react to the presence and concentration of volatiles.

Table 3: Physiological indices of maturity during ripening of western cantaloupes *cv. Mission*, and green-flesh *cv. Makdimon*.

Ripeness Stage (expert)	External Color	Stem Detachment (lbs)	Flesh Firmness (lbs)	Total Soluble Solids (%)	Ethylene $\frac{\mu l}{Kg \cdot h^{-1}}$	Aromatic Volatiles $\frac{\mu g}{Kg \cdot h^{-1}}$	Electronic Sensor (mV)
Data from 83 fruit, <i>cv Mission</i>							
Unripe	Green	13.1 a ¹	11.3 a	14.2	0.622	0.00824	85 a
Half-ripe	Yellow-Green	2.9 b	8.2 b	14.8	1.617	0.81	331 b
Full-ripe	Yellow	1.6 b	4.4 c	13.2	1.314	2.07	693 c
Over-ripe	Yellow-Orange	1.3 b	3.6 c	12.5	0.854	2.69	983 d
Data from 16 fruit, <i>cv Makdimon</i>							
Unripe	Yellow-Green	16.1 a	2.27 a	12.3	0.341	²	129 a
Half-ripe	Yellow	10.2 ab	2.68 a	13.6	0.532	-	204 b
Full-ripe	Yellow-Orange	6.2 b	2.50 a	14.4	0.702	-	277 c
Over-ripe	Yellow-Orange	6.2 b	2.30 a	13.1	0.818	-	303 c

There was a decrease in the stem detachment force and in the firmness of internal flesh with increasing fruit ripeness. Ethylene production and TSS both reach peak values at an intermediate ripeness stage, and then decrease with continued ripening. In contrast, aromatic volatile production increases throughout ripening. The electronic sensor readings are strongly correlated to the total amount of headspace gases ($R^2 = 0.87$). Linear regression of the sensor readings with the different compounds being treated as individual inputs results in a significant coefficient of determination ($R^2 = 0.975$), indicating that the sensitivity can vary for specific volatile compounds.

To demonstrate the ability of the sensor system to determine ripeness one must show a correlation between the output of the system and the actual ripeness stage of the fruit. While most research efforts have selected ripeness indices which are presumed to have a consistent relationship with ripeness, our system can be tested directly on the best axis by which ripeness can be measured - the time axis. Eight mature green fruit of two different varieties - the California 'Mission', and the midwestern 'Superstar' were selected. These 8 fruit were sampled by the sensor while attached to the vine on alternate days throughout a period of two weeks.

¹Mean separation among ripeness stages by Duncan's range test ($P \leq 0.05$)

²Insufficient data available.

Because each fruit was not at exactly the identical ripening stage when initially selected, the response for each fruit was shifted along the time axis based on the timing of appearance of the abscission layer. Our study not only demonstrates the ability of the system to determine ripeness, but also the feasibility of using it as a field operable technique, since it was unaffected by environmental conditions. The response of the electronic sensor in the lab and in the field are strongly correlated (R^2 ranging from 0.8 to 0.93 depending on cultivar), and are partly due to the time lapse between the measurements, and the environmental changes such as temperature, since fruit were measured in the field while attached to the vine immediately before harvest, and 24 hours later in the lab. We expect further improvements in the field response by modifying the sensing head for better contact.

Figure 10. Typical dynamic response of electronic sensor to various fruit ripeness stages.

The electronic sampling of volatiles appears to present a highly feasible technique for nondestructive sampling of melons. The ability to discriminate between ripeness stages was higher than all other destructive and non-destructive techniques applied, for all cultivars tested. The effect of climate and environmental parameters did not seem to prevent differentiation between ripeness classes, under the range of the existing climate during which tests were made outdoors. Fruit surface temperatures varied from 18°C to 41°C, as data were collected at randomly different hours of the day. Relative humidity levels varied from 37% to 85%.

In order to increase the accuracy of the prediction of ripeness, further studies are required, including the effect of temperature for measurements taken under extreme conditions, and also the effects of storage duration. Results from such studies may be useful as a device for estimation of predicted shelf life and optimal consumption date, assuming future storage conditions are known. The applicability of this system to be used for a wide range of fruit, in particular those fruits which present no obvious external indices of ripeness exists (e.g. watermelon) has not been yet tested, though some preliminary tests with Golden Delicious apples indicates a strong correlation and appears most promising.

2.1.5.2 Melon Ripeness Determination by Image Features

In order to determine relationships between ripeness and visual features, observations were made on *cv. Superstar* and *cv. All-star* melons grown at the O'Neal farm south of Lafayette, Indiana (Cardenas-Weber, 1991). Twenty four melons were selected to represent unripe, half-ripe, and ripe melons, as classified by an expert observer. The results are shown in Table 4. Slip is defined as the ease of separating the stem from the melon and is a commonly-used criteria for ripeness in the Midwest. Sample 7 was harvested unripe and stored in a refrigerator for 2 weeks before analysis; thus it is listed as "old. The sugar content was determined by squeezing a sample of melon and measuring the solids in a refractometer. Color, venation, and ribbing were qualitative observations.

From these data it can be seen that the sugar content of muskmelons is proportional to ripeness as classified by slip and an expert's judgment, but only at the time of harvest. Sugar content increases with time in storage without change in external appearances of the fruit. As muskmelons ripen, the exterior color changes from green to light yellow. Early in the season, the fruit are dark green with almost no venation pattern, or texture in the image. In the unripe stage, the melons have a "bluish" cast and the venation pattern has deepened and the vein color has become lighter and appears as an intricate network of very light yellow, raised lines. This change in the venation pattern is difficult to quantify. In full ripe melons, the venation pattern is less pronounced and the melon surface is smoother. It may be more practical to quantify the color difference between the vein and the background melon surface. In the half-ripe stage (half slip), the color changes to very light yellow. In the full-ripe (full slip) stage, the melons appear as deep yellow. For melon varieties with longitudinal ribs, light and dark patterns appear as the fruit ripens, but at full-ripe, the color is indistinguishable. Although these color changes are perceptible to a trained expert, the numerical differences are slight and will be difficult to quantify. Also, the absolute levels are functions of many factors, including the intensity and quantity of solar radiation received during the growing season. Thus, color imaging will be at best a relative indicator of muskmelon ripeness.

2.1.6. Summary of Sensor Developments

Different approaches were investigated to detect the melons locations: gray-level and color based vision, laser based profiling system, and Near infrared imaging. A summary of results concerned with the first three approaches is presented in Table 5. Most of the melons (85%) are successfully detected.

Results indicate that both the gray-level and color-based algorithms have equivalent performances the tested cultivars. The fruit dimension (the ratio between the fruit and its leaves diameters) is the critical parameter: detection results improve for bigger fruit). The present gray-level based algorithm uses the reflectance, shape and size information of the objects of interest. A global thresholding technique is applied for segmentation. An adaptive segmentation technique must be developed. This can be achieved by a local independent segmentation of the image regions which are suspected to be ripe fruits; these regions must be determined preliminary based on their high average intensity. The algorithm's execution time is acceptable since it is in the range of the estimated average robot harvesting cycle (Edan and Miles, 1991). The color-based approach must be further investigated. The problem is that spectral difference between melons and leaves is variety dependent and sometimes leaves

appear yellow similar to a ripe fruit what can be a reason of false detection. High quality color camera should be used to improve images acquisition; colors must be calibrated. Color segmentation technique can be applied in future. Color-based and (shape-based) gray-level approaches can be combined to improve the detection results. For this, sensor fusion techniques must be developed. Artificial lighting and shading effects must be further studied. The impact of the air blast is significant and it's interaction with the vision requires more investigation. Hardware synchronization of the vision and the air blast activity must be developed. The following must be done to improve the location performance of the laser system: the algorithm execution time must be decreased; analysis of a number of profiles obtained from the same melon must be introduced; interaction of the laser system with the blower must be studied. Additional research is needed to evaluate the potential of Near infrared imaging for melon location. Combining Near Infrared imaging with regular vision can improve melon detection results.

Table 4. Ripeness observations for muskmelons based on qualitative visual features.

Sample	Slip	Ripeness	Color	Visual Features	Sugar (° Brix)
1	full	ripe	dark yellow	1 mm veins light green	7.7; 7.9
2	full	ripe	very yellow	deep veins light- green	9.9; 9.7
3	half	about ripe	light yellow	deep veins, full netting	5.3
4	half	unripe	light yellow	green ribs, light veins	4.8; 4.4
5	no	unripe	dark green	deep ribs and netting	6.7
6	no	very unripe	light green	deep veins, light- green	5.5; 4.2
7 (old)	no	very unripe	light green	---	8.1
8	full	ripe	very yellow	light green	7.4; 6.8; 6.6
9	half	about ripe	light yellow veins	---	4.7; 5.2
10	no	unripe	very green	light veins	6.5; 8.1
11	no	unripe	green	full netting	4.7; 5.1; 6.0
12	full	ripe	dark yellow		8.5; 7.8; 7.6

Table 5. Summary of Real-Time Vision and Laser System detection results

	Image Processing System		
	Gray-Level	Color	Laser
Detected Melons	82 - 86%	82-88%	74
Execution Time (sec/image)	1.1	1.3	5.5

2.2 Intelligent Control of a Field Crops Robotic Machine

The uncertainties in melon location, size, shape, and maturity require a perceptive, programmable and intelligent machine. The harvester must identify fruit that are partially occluded, decide whether a specific fruit is ripe or not, and pick just the ones which meet ripeness and quality criteria. Uncertainty in location caused by variable ground speed and uneven terrain requires velocity and position monitoring. The only machine capable of performing such actions is a robot.

Many robot applications require the robot to function under dynamically changing conditions. The robot must operate in a changing environment and generate the necessary responses as fast as possible. For example, in object tracking, the goal of the manipulator is to grasp an object located on a moving conveyor. The object's position continuously changes and the manipulator must update its trajectory accordingly.

Another example of a dynamic task is the situation when only partial information about the target location is known *a priori* or if the information is not accurate enough. If the exact position of the target can be determined only at the end of the trajectory, the robot must begin the movement without knowing the final end-point of the trajectory and must update its trajectory after it receives additional information about the target. The change in target position either due to the physical movement of the target or due to additional position information provided later is denoted here as a target switch.

All these examples require continuous updating of the ongoing trajectory due to possible target switches. Such trajectory modifications must be performed very quickly, *on the fly* without interrupting the current movement. Trajectory planning generates a sequence of points such that the robot arm must pass through each of them sequentially at a specified time. Movement of the robot arm between two neighbor points is usually determined and controlled by a low-level robot controller and cannot be controlled interactively by user-written control software.

The problem of trajectory modification can be solved in different ways using alternative approaches and mathematical tools. Every method has advantages and disadvantages. The objective of this work was to design a real-time algorithm for the trajectory modification based on a scheme which generates a smooth trajectory and minimizes the duration of the movement. This scheme is called superposition.

2.2.1. Superposition Scheme for Real-Time Trajectory Modification

2.2.1.1. Description of the Superposition Scheme

Consider the situation when a robot manipulator is moving toward a static target and this target becomes irrelevant because a new target appears, or when the robot moves toward a moving target. The goal of the control program is to modify the current trajectory and to provide a new trajectory in order for the robot to reach the new target position. Moreover, the whole trajectory from the beginning of the movement to its end must satisfy acceleration and/or velocity constraints. One way to change the trajectory is to abort the rest of the trajectory and start a new one toward the second target. Of course, switching between the two trajectories should be as smooth as possible. An alternative is to incorporate the old

trajectory in some way into a new one which moves the robot arm toward the location of the second target. This overlapping of the old and new trajectories is called superpositioning. This idea resulted from the study of human motor behavior when a target-tracking task is performed and the trajectory must be modified in response to a sudden change in the target location. These studies demonstrated the following observation which has provided the basis for the superposition strategy.

Each unit of the trajectory is computed and executed independently of the other one. The first unit is based on the original-position-to-first-target vector, while the second one is based on the first-target-to-second(new)-target, and not on the current-point-to-second-target vector. Therefore, what actually happens in the target-tracking tasks performed by humans, is the following: the trajectory between the initial location and the first target is computed based on the displacement vector connecting the initial position and the first target and performed entirely from its beginning to its end. Upon the appearance of the second target the trajectory connecting the first and second targets is computed. The planning of this second unit of motion is based on the displacement vector connecting the first and the second targets. The second unit is performed by the end-effector simultaneously and independently from the first one. Thus, the actual trajectory carried out by the end-effector is the vector sum of the two trajectory units with a correct delay.

This strategy has three important advantages:

1. The modification of the trajectory can take place and be completed independently of the time of switch. Hand location, velocity and acceleration at the time of the trajectory modification may be unknown, since they are not required for the computation of the new trajectory.
2. The superimposed trajectory units can be planned in parallel. Therefore, to plan the elementary motions it is possible to use a multiprocessor system, which permits a high degree of parallelism both in planning and executing the task.
3. If both trajectories are smooth, so is their sum, and the resulting motion is both smooth and achieves the desired values of position, velocity and acceleration at the end point of the trajectory unit.

2.2.1.2. The Mathematical Model

Let *Start*, *End* and *New* be the initial, end and new targets, respectively. The movement starts from the *Start* target to the *End* one and at some intermediate point the *New* target appears. At this point of the trajectory, the *End* target becomes irrelevant and the current start-to-end movement must be changed to ensure that the robot arm intercepts the *New* target.

According to the superposition strategy we should compute a vector sum of two trajectories: the Start-to-End trajectory and End-to-New trajectory. The sum should give us a smooth trajectory which will lead to the New target and should also not violate any constraints. However, if both trajectories satisfy given constraints, this does not imply that the vector sum satisfies the constraints. Therefore, the summed trajectory may violate the given velocity or acceleration constraints. Even if the summed trajectory does not violate the constraints it might not be optimal with respect to time. Thus, the problem is to find the time of switching from the original trajectory to the new one *S* which provides an optimal trajectory that minimizes movement time and does not violate any kinematic constraints.

Let T_1 be the time it takes to move from *Start* to *End* and T_2 be the time of movement from *End* to *New*. Let also S_1 be the time when the *New* target is discovered and S be the time when the switch occurs. In the study involving biological motion each unit was found to be the minimum jerk hand trajectory between two end points. This trajectory is described by the following 5th order polynomial:

$$X(t) = X_0 + (X_f - X_0) \cdot 6\left[\left(\frac{t}{T_1}\right)^5 - 15\left(\frac{t}{T_1}\right)^4 + 10\left(\frac{t}{T_1}\right)^3\right] \dots\dots\dots [8]$$

where T_1 is the duration of the movement.

This expression satisfies the initial position X_0 , zero initial velocity and acceleration, zero final velocity and acceleration and final position X_f . For the second movement which starts at some time S and has zero initial position, velocity and acceleration, the expression becomes:

$$X(t) = A_2 \cdot \left[6\left(\frac{t-S}{T_2}\right)^5 - 15\left(\frac{t-S}{T_2}\right)^4 + 10\left(\frac{t-S}{T_2}\right)^3\right] \dots\dots\dots [9]$$

where A_2 is the amplitude of the movement.

The velocity of the movement can be derived by differentiating Equations 8 and 9:

$$V_1(t) = X_1'(t) = A_1 \cdot \left[30\left(\frac{t}{T_1}\right)^4 - 60\left(\frac{t}{T_1}\right)^3 + 30\left(\frac{t}{T_1}\right)^2\right] \dots\dots\dots [10]$$

$$V_2(t) = X_2'(t) = A_2 \cdot \left[30\left(\frac{t-S}{T_2}\right)^4 - 60\left(\frac{t-S}{T_2}\right)^3 + 30\left(\frac{t-S}{T_2}\right)^2\right] \dots\dots\dots [11]$$

Each motion unit has a bell-shaped velocity profile. The summation of two motion units with such velocity profiles results in the profile represented in Figure 11. In the figure the solid line stands for the summed trajectory and the dashed line stands for the separate trajectory units. Both units have equal amplitude, S is approximately equal to $T_1/2$.

Now consider the summation of the two trajectories with the same amplitudes and the same durations and with $S_1 = S = 0$. In this case, the peak of the velocity of the resultant trajectory is twice as large as the one for the each motion unit. Both units are performed with the maximal permitted velocity and therefore the summed velocity will violate the given velocity constraints. Thus, whether the final velocity profile of the motion violates the constraints or not, is strictly dependent on the choice of the switching point S . The solution is to pick S and T_2 such that no limitation will be exceeded after the superposition of the first and second motion units. Since T_1 is chosen ahead of the movement and kept fixed during the entire motion, it makes sense to choose it to be the smallest possible value that does not violate the constraints. S for the second movement can be chosen to be equal to T_1 . This will assure that the combined velocity will not violate the constraints (in this case the units are performed consequently). But there probably exists a smaller S which also avoids violation of the constraints and makes the whole movement duration smaller. Another possibility is to pick T_2 smaller. That implies a smaller maximal velocity of the second unit and permits adding the unit with a smaller S , and results in a smaller traveling time.

The sum, $T_2 + S$, defines the end-time of the second movement and the goal is to minimize it, while not violating any constraints. But the final traveling time for the whole motion might not be equal to the time for the second unit ($T_2 + S$). The second motion can be completed while the first one is still in progress. In this case the total motion time will be equal to the fixed T_1 . Generally, the total traveling time is equal to the maximal value between the duration of the first and second motion.

Therefore, the goal is to find T_2 and S that satisfy:

$$\max[\min(T_2+S), T_1], \text{ where}$$

$$(T_2, S) \in \{[0, \infty] \times [S_1, \infty] \mid \text{Check}(S, T_2) = \text{TRUE}\}$$

Here $\text{Check}(S; T_2)$ is a check-function which has the value TRUE, if the sum of the first trajectory and the second one, summed at the switching point S with time T_2 for the second trajectory does not violate constraints, and FALSE otherwise. This function implicitly incorporates the given velocity and/or acceleration constraints. In this work the superposition algorithm has been implemented for velocity constraints. However, the suggested method can be used for any type of constraints which could be incorporated into the check-function.

Figure 11. Velocity profile for the addition of two motions.

2.2.1.3. Real-Time Implementation of the Superposition Strategy

To be used in real-time systems, the superposition algorithm must be fast because recalculation of the trajectory is usually performed *on the fly*, without interrupting the robotic motion. A new, fast algorithm for determining the optimal $(T_2 + S)$ is necessary to apply the superposition scheme for the real-time trajectory planner. The concept of this algorithm is to define some domain where the optimal switching time S can be found, reduce this domain as

much as possible to decrease the number of the points which must be checked and try every remaining candidate in this domain to find numerically the best (optimum) S.

Attempts at finding an analytical solution to this problem by applying optimal control theory resulted in the derivation of a system of nonlinear equations, which should give the optimal solution but requires a very complicated numerical procedure. Previous studies have shown that for a single motion, the maximum velocity is $\frac{30 \cdot A}{16 \cdot T}$, where A is the amplitude of

the trajectory and T is the time of the movement. Therefore, $T_1 = \left| \frac{30 A_1}{16 T} \right|$ is the minimum value of the time for the first movement, where $V_{\max}$ is the maximum permitted velocity.

In this model, the amplitude of the *Start-to-New* trajectory was chosen to be 100 cm and the maximum permitted velocity was selected to be 187.5 cm/sec. These values are convenient since they imply $T_{i\min} = \frac{|A_i|}{100}$ and the minimal time $T_{i\min}$ for the first trajectory is equal to 1 sec. Amplitudes of the first and second motions are denoted as A_1 and A_2 correspondingly.

Now, consider the behavior of the check-function depending on the amplitudes of the trajectories. This function is very important because its True values define a real-time search (RTS) domain to look for $\min(S + T_2)$. To analyze the domain, a special simulation program in C was written. The program receives the coordinates of the targets and starts a loop where for every s ($0 \leq s \leq T_1$) and t ($T_2 \leq t \leq 3 \cdot T_2$, or $0.5 \cdot T_2 \leq t \leq T_2$), (the choice of these intervals is explained in the next section) and a step Δt , the check-function is calculated.

Then the output of the program is represented as a plot of the check function on a Cartesian coordinate system where the abscissa X represents the switching time s and the ordinate Y axis represents the minimal time for the second trajectory t. The True values of the check-function are plotted by the gray color and the False values by the white. Thus, the gray area on the plot shows the distribution of a desired domain.

For discussion of the details of the domain's structure, the problem is separated into three cases depending on the signs of the amplitudes:

1. $\text{Sign}(A_{1x}) = \text{Sign}(A_{2x})$,	$\text{Sign}(A_{1y}) = \text{Sign}(A_{2y})$
2. $\text{Sign}(A_{1x}) = -\text{Sign}(A_{2x})$,	$\text{Sign}(A_{1y}) = -\text{Sign}(A_{2y})$
3. $\text{Sign}(A_{1x}) = \text{Sign}(A_{2x})$,	$\text{Sign}(A_{1y}) = -\text{Sign}(A_{2y})$

Case 1: $\text{Sign}(A_{1x}) = \text{Sign}(A_{2x})$, and $\text{Sign}(A_{1y}) = \text{Sign}(A_{2y})$. S and T_2 must meet two trivial conditions:

$$0 \leq S \leq T_{1\min}$$

and

$$T_{2\min} \leq T_2 \leq (T_{2\min} + T_{1\min})$$

The condition $S > T_{1\min}$ corresponds to two separate motions carried out sequentially and obviously does not violate the constraints. Increasing S doesn't improve the results as well. Taking $T_2 > (T_{2\min} + T_{1\min})$ does not provide any improvements because S can always be chosen to be equal to $T_{1\min}$ (this is the worst case) and then the time of the final movement will be equal to $T_{2\min} + T_{1\min}$.

Figure 12 represents a model with both amplitudes (of the first and the second trajectories) greater than zero. The gray area represents the RTS domain. The RTS domain

consists of two parts where the left one becomes smaller as the ratio between the second amplitude and the first one increases (A_1 is kept fixed). Such behavior corresponds to the situation when the second amplitude is relatively small and the second movement starts and ends within the first half of the first movement without violating the constraints. For larger A_2 , such summation of the trajectories is impossible and this part of the RTS domain disappears.

The minimum of the linear function $S + T_2$ is found on the boundary of the RTS domain. This is obvious: suppose the minimum of $S + T_2$ is located not on the boundary but inside the domain. Then there is such S and T_2 that $S^* < S$ and $T_2^* < T_2$ and the point $(S^*; T_2^*)$ belongs to the domain. It means that the minimum is not in $(S; T_2)$ but in $(S^*; T_2^*)$ which results in a contradiction.

Given such A_1 and A_2 that a left (small) part of the RTS domain exists and for a small enough S_1 then for $S = S_1$, then T_2 can be chosen:

$$T_2 = \min(T_2^*), \text{ where } (T_2^*, S_1) \in \text{RTS}$$

If the two trajectories starting at the switching time S_1 cannot be summed, then the minimum will occur on the tangent line $S + T_2 = C$ to the RTS domain where C is some constant. The algorithm for finding the tangent point between the line $S + T_2 = C$ and the boundary of the domain will be described later.

Case 2: $\text{Sign}(A_{1x}) = -\text{Sign}(A_{2x})$, and $\text{Sign}(A_{1y}) = -\text{Sign}(A_{2y})$. A_2 is less than zero, while A_1 is still fixed and positive. In this case, condition (1) for S still holds. However, for T_2 a slightly different inequality exists:

$$0 < T_2 < T_{2\min}$$

This follows from the fact that the velocity vectors for the first and second movements have opposite directions and when summed will compensate for each other. It means that the absolute value of the velocity for the second movement can be chosen to be greater than the maximal permitted velocity because being added to the first movement it will be compensated for and the final trajectory will still not violate the constraints. Hence, the value of T_2 can be decreased and be smaller than $T_{2\min}$.

On the other hand, T_2 must not be chosen to be too small because the maximum velocity of the second motion will be too high and adding the first movement will not keep the robot within the constraints. It is quite clear that the low bound for T_2 should be taken to be equal to $0.5 T_{2\min}$. The value $T_2 = 0.5 T_{2\min}$ implies $V_{2\max} = 2 \cdot V_{\max}$ which is still compensated for by $V_{1\max}$ which has opposite direction and the resulting velocity will be less than $V_{\max}$. But a smaller T_2 results in an excessive maximum velocity for the second movement and consequently violates the velocity constraints after the trajectory superposition.

Figure 12 represents the search domain for cases of different amplitude signs with velocity constraints in effect. On this plot the upper bound for T_2 is chosen to be slightly greater than $T_{2\min}$ and one can see that all points where $T_2 > T_{2\min}$ are in the RTS domain (gray strip in the upper part of the plot). The drop-shaped part becomes smaller when the absolute value of $A_2 = A_1$ increases. The minimum of the $(S + T_2)$ expression are found on the tangent line $S + T_2 = C$ of the RTS domain.

Case 3: $\text{Sign}(A_{1x}) = \text{Sign}(A_{2x})$, and $\text{Sign}(A_{1y}) = -\text{Sign}(A_{2y})$. Two-dimensional movements could be considered as two separate one-dimensional movements combined together. Each of these movements must obey its own constraints and the minimal time of the whole movement will be defined by a stronger condition on $T_{\min}$ and $T_{\min}$. The case of amplitudes with equal signs implies $T_{2\min} \leq T_2 \leq 3 \cdot T_{2\min}$ and the case of different amplitude signs implies $0.5 \cdot T_{2\min} \leq T_2 \leq T_{2\min}$. Thus, the part of the trajectory which has equal signs both for A_1 and for A_2 has also longer duration, and the whole time of the movement will be defined by the time of this particular motion component.

Figure 12. RTS domain for different $A_2 = A_1$ ratios.

2.2.1.4. Optimal Switch Time in the Case of Velocity Constraints

An experiment was performed with velocity constraints and an algorithm developed to search the domain represented in Figure 12. The first step of the algorithm is to define the signs of trajectories and to decide which type of search domain to consider. Depending on the results, the algorithm branches into two parts and starts the appropriate procedure. Each time the superposition algorithm is started only one procedure is invoked. The algorithm generates optimal S and T_2 , but the whole time of the movement is defined as $\max(S + T_2; T_1)$. It means that the time of movement cannot be made less than T_1 even if S and T_2 are small. If for some particular case $(S + T_2) \leq T_1$ it makes sense to increase the time T_2 for the second trajectory and finish the second movement at the same time as the first one. This approach allows us to perform the second movement more smoothly and doesn't increase the final time for the whole movement as well. Thus, if as a result of computation $(S + T_2) \leq T_1$, the value of T_2 is set equal to $T_1 - S$.

2.2.1.5. Optimal Switching Time for Equal Sign Amplitudes

The first step of this procedure is to check whether the second trajectory can be performed within the first half of the first trajectory. This corresponds to the case when the left small part of the RTS domain exists (part A in Figure 30) and S_1 is small enough. As mentioned before, S_1 is the time when we are informed about the appearance of the new target, S is the optimal switching time we are looking for ($S_1 \leq S \leq T_1$) and T_2 is the optimal time of motion ($T_{2min} \leq T_2$). The algorithm tries to find the switching point S and duration T_2 in the part A if the following two conditions hold: $S_1 < 0.25 \cdot T_1$ and $(\frac{A_2}{A_1}) < 0.4$. If one of

these conditions fails, then either part A doesn't exist or S_1 is too large and A is useless. If both conditions hold the algorithm runs a simple numerical procedure to find the point $(S; T_2) \in A$. The algorithm initiates S to be equal to S_1 and tries to find the first possible value T_2 . If the optimal point T_2 is found in the above algorithm the second trajectory can be superimposed immediately, ($S = S_1$), and the whole procedure finished.

If under the given conditions part A of the domain doesn't exist or S_1 is too large then the algorithm starts processing part B of the domain. First, point S^* is found. Obviously for $s > S^*$ the second trajectory can be superimposed with $T_2 = T_{2min}$ and the constraints are not violated. S^* is the right boundary for the domain searched for the optimum S and T_2 . This means that only $s \leq S$ are considered.

The procedure used to find the right boundary S^* is based on a binary search algorithm. The program divides the search region from S_1 to T_1 into two halves and depending on the value returned by *Check-function* applied at the middle of the search interval, chooses one of the halves as a new search region for the next iteration. This is a fast algorithm for performing binary searches. After N iterations, the search interval is reduced to $\frac{1}{2^N}$ of the original range of uncertainty.

After the point S^* is found the algorithm progresses along the boundary of the domain and calculates the sum of s ($s \leq S^*$) and t corresponding to it (Figure 13). The boundary value t is calculated using the same binary search as described above. That is, for every s the binary search algorithm searches in the region from T_{2min} to $3 \cdot T_{2min}$ and produces t corresponding to the given s .

The sum of s and t monotonically decreases to a minimum value. The s and t which imply the minimum sum are the desired optimum parameters of the trajectory superposition. When the sum of these values begins to increase while the algorithm goes from S^* to S_1 , the algorithm stops and returns S equal to the current s and T_2 equal to the current t as the optimal switching time and duration for the second trajectory. This criteria is successful because the function $s + t$ for s and t corresponding to the boundary has only one global minimum.

2.2.1.6. Optimal Switching Time for Different Sign Amplitudes

If the amplitudes of the two motions have opposite signs, the RTS domain has a completely different shape as illustrated on the right part of Figure 13. The domain of this *drop-shaped* form is handled by another procedure which is the second part of the algorithm.

As previously mentioned, the optimal time for the second movement can be made even less than T_{2min} . This can occur if the second movement starts in the first half of the first movement and its amplitude is relatively small. The algorithm for finding the optimum switching time in this case is also based on the binary search algorithm used previously for boundary detection. Since the minimum possible switching time is S_1 , the algorithm starts processing from $s = S_1$. For every $s \geq S_1$, the corresponding t is calculated. The point $(s; t)$ is located exactly on the boundary of the domain and is calculated in the same way as for the previous case. The sum of s and t is calculated and the algorithm terminates when the minimum is found (*i.e.*, when the sum stops decreasing).

Figure 13. Finding the optimum solution using the RTS domain.

2.2.1.7 Complexity of the Superposition Algorithm

The complexity of the superposition algorithm will determine the speed with which it can be executed, and thus determine its suitability for real-time control. The time that the algorithm takes to find an optimal switching point is strictly dependent on the control interval Δt . Define the RTS domain on the set $(0; T_{1min}) \times (T_{2min}; 3 \cdot T_{2min})$, and suppose the space has dimension $n \times 3m$. That is, there are n intervals in $(0; T_{1min})$ and m intervals in $(0; T_{2min})$. For the purpose of the complexity analysis the constants are not important and can be removed.

The dimension of the space as $n \times m$. Obviously, for smaller time steps Δt , the numbers of the points n and m are larger. The final estimation of the algorithm complexity is:

$$C_{alg} = O(m^2 + m \cdot \log n + n \cdot m \cdot \log m) \dots\dots\dots [12]$$

2.2.1.8. Implementation of the Superposition Algorithm

The superposition algorithm was tested on an Adept-One Robot and the prototype robotic harvester for melons. The Adept-One Robot was used for conducting the experiment with moving arm approaching a static target. The experiment with a target moving relatively to the robot was conducted on the agricultural prototype.

2.2.1.8.1 Adept-One System Overview and Description of the Experiments

The superposition scheme for motion planning was implemented on an Adept-One robot. Visual targets were detected with an OptoTrak infrared machine vision system. A diagram of the physical set-up is shown in Figure 14.

First the system is initialized, which requires downloading the control software into the OptoTrak transputer and the Adept-controller. Then the program sends the robot arm to the initial position which is chosen to be marker number 1. The program queries the OptoTrak system about the location of the first marker and commands the manipulator to move to this location. When the motion terminates, the program waits for user's input which should be the number of the next trajectory marker. When the executive controller is notified of a new target (by receiving the number of the next trajectory marker), the program obtains target coordinates from the OptoTrak, computes a minimal jerk trajectory from the current position to the second target, and guides the robot arm according to the new path coordinates.

During the robot arm motion, the program waits for another target number from the operator. Once it is notified of the new target location, the program starts the superposition algorithm and updates the current trajectory according to this scheme. Depending on the calculations, the robot arm can either proceed with the current trajectory all the way to the calculated switching point or switch to the new trajectory immediately. During the motion, the program is ready for the user's input of a new target. If this input is received, the input number becomes the number of the new target (marker) and the program moves the robot arm to this target calculating the new trajectory according to the superposition algorithm in real-time.

The Adept-One manipulator is a four degrees of freedom direct drive arm. The two links are rotational with parallel vertical axes and thus work in the horizontal plane. The third axis (quill) is a translational one and has a vertical motion. The fourth axis is the rotation of the entire quill around a vertical axis. In the current experiment, the third and fourth axes were not controlled. The vertical position of the end-effector was kept constant.

The language used for programming the Adept-Controller controller is VAL-II. This language is interpreted, not compiled into robot commands and therefore is extremely slow. Only position and partial velocity control are permitted. The controller accepts and responds to Cartesian or joint space position commands. During these experiments, only Cartesian coordinates were used. Software written in VAL-II is very slow and not suitable for computationally intensive and real-time applications such as superposition trajectory control. Thus, all position computations must be performed on a computer external to the Adept-One

Robot. The external control program outputs the position to move to through a serial interface to the Adept-One controller for execution.

The main hardware for communication and computation was a PC-AT 486 (33 MHz) with an internal mathematical coprocessor. The OptoTrak interface adapter card was added for communication with the OptoTrak transputer. Communication with the Adept-One controller was performed using a standard serial interface RS-232.

The OptoTrak machine vision system was used to determine the target's 3-D position. The target is an infrared emitting diode called a marker. The infrared light emitted by the marker can be detected by infrared sensitive cameras. The 3-D position is transmitted to the PC. The accuracy of the calculated position is very high and the time required for these calculations is negligible.

Six markers were placed on the Adept-One Robot's work table. These markers served as six different targets in the movement control scheme. One additional marker was placed on the end-effector of the robot and was used for reading the actual position of the end-effector as it varied with time. These data were used later to examine the actual velocity profiles during the arm movement.

Figure 14. A diagram of the experimental setup

2.2.1.8.2. Results for the Adept-One Robot

As was shown in Equation 12, the superposition algorithm performance is directly dependent on the number of points which divide the amplitude curve. These are numbers m and n in the complexity formula. For these experiments, m and n are defined as follows.

The maximum distance between markers (targets) was approximately 100 cm. The maximum permitted velocity for both the X and Y axes was 10 cm/sec. Thus, the maximum time for every target-to-target motion is $\frac{30 \cdot A}{16 \cdot T} = \frac{30 \cdot 100}{16 \cdot 10} \approx 18.7$ sec. The time of the minimum movement segment Δt was taken to be equal to 0.13 sec. This implies the maximum number of points n to be equal to 144. This is the number of points along the S axis ($0 \leq S \leq T_1$). The step along the T_2 axis was chosen to be the same and the number of points m was set equal to 144.

To generate a movement between two points the controller needs 0.032 sec. Thus the superposition algorithm has approximately 0.1 sec to find the optimal switching time and replan the current trajectory. For the experimental conditions used in this study, 0.1 sec turned out to be the minimum time required for the program running on a 33 Mhz 486 personal computer to finish the calculations and modify the trajectory before the next point was requested by the controller. By executing the superposition algorithm within 0.1 seconds, the personal computer was able to feed the Adapt-One Robot a stream of path points, making the motion smooth and continuous.

Decreasing Δt causes an increase in the numbers of points n and m and a subsequent increase in computation time. On the other hand, if the value of Δt is decreased substantially, when a new target appears and the superposition algorithm is started, it may not be able to find the optimum switching time in the $t=0.032$ seconds available, and the Adapt-One Robot controller must wait for the next sequential trajectory point. As a result of this delay, the robot stops and the motion is interrupted.

Figure 15 shows the trajectory of the arm when only one switch of the target occurred. The *Start* (initial) target is marker number 1, the *End* target is marker no. 6, and the *New* target which the robot might switch to is marker no. 4. The *New* target was given to the program when the robot had covered half of the distance between the first and sixth markers. But the switch of the trajectory occurred only at the point B which is about $3/4$ of the distance away from the *Start* target. Thus, on the A-B motion segment only the first movement was performed. At point B, superposition was applied and during segment B-C both motions were active. During the motion along this segment, the robot changed the direction of the movement smoothly. The first movement was completed when the second one was still in progress (point C) and as a result the last segment C-D of the trajectory was a straight line which corresponded to the second movement.

Figure 16 represents the velocity profile for the path shown in Figure 15. Note that the velocity profile was derived from two bell-shaped profiles summed together starting at the switching point. This point corresponds to the point B on the plot. This is the actual velocity profile obtained by measuring at every control interval Δt the position of marker number 0 placed on the robot arm.

Since the velocity of the robot arm cannot be controlled directly, it must be approximated by using short path segments which have a constant velocity. For this reason, the actual velocity profile was found to be less smooth than the mathematical model.

Uncertainties in sensing marker location and arm position (OptoTrak measurements) also contributed to jittery paths.

Multi-switch movement is represented in Figures 17 and 18. The *Starting* target is again at marker number 1 and the sequence of the new targets is 2, 4, 6 and 2. Assume that knowledge about *New* targets is incremental. That is, when the information about the *New* target arrives, none of the previous units are changed. T_i and S are computed for the new trajectory and added on top of the existing ones (superimposed), while making sure not to violate any constraints.

If the switch is made at the end of the trajectory (for example near marker four), where the velocity is low, the radius of the turn is small. Slow speeds permit sharper turns without increasing the jerk. On the other hand, if the switch occurs at the middle of a trajectory, when the speed is close to its maximum (for example on the segment located between marker four and six), then the turn generated by the algorithm has a larger radius and the change in the movement direction is performed more gently.

On the velocity profile, the local minimum of velocity A occurred during the sharp turn near marker four, while the local minimum B occurred during the smooth trajectory turn near marker six.

Figure 15. Trajectory of the movement with only one switch.

Figure 16. The velocity profile for the one-switch movement.

Figure 17. Trajectory of the multi-switch movement.

Figure 18. Velocity profile for the multi-switch movement

2.2.1.8.3. Results for the Robotic Melon Harvester.

The superposition algorithm was tested also on the robotic melon harvester. The harvester consists of a robot arm mounted on a mobile platform which is drawn by a tractor along the melon bed. The platform is a rectangular steel frame with a 3-axis Cartesian robot mounted on it. The manipulator for each axis is controlled by a Galil DMC-630 controller which is connected to a 33 Mhz 486 PC. The robot is supplied with two visual sensors (CCD cameras). One camera is mounted on the frame ahead of the robot and the other is mounted on the manipulator.

When dealing with industrial controllers such as the DMC-630, there are several limitations which must be considered by the trajectory planner. The most important one is the default velocity profile of the robot motions. When the DMC controller moves the manipulator from one point to another, it uses a trapezoidal profile as shown in Figure 19 instead of a bell-shaped velocity profile. To construct this type of movement, the controller uses two constraints: maximum velocity and maximum acceleration. Typically, acceleration is limited by the torque or power capability of the servo-actuator and depends on the mass of the load being moved. Maximum velocity is limited by the peak speed of the actuator and is affected by drive mechanics and friction. In the trapezoidal profile, the arm increases speed (segment AB) with the maximum possible acceleration up to the maximum permitted velocity (V_0), then moves with constant velocity (segment BC), and finally decelerates (segment CD). With this approach, the final acceleration and velocity of the arm are zero. The velocity of the motion can be changed while the motion is in progress by a command issued by the PC to the DMC controller. The only way to obtain a bell-shaped velocity profile is to

Figure 19. The trapezoidal velocity profile.

constantly update the velocity by issuing the proper commands to the DMC. This requires extensive communications between the computer and the controller which slows down and even blocks other processes. Hence, this is not a feasible approach.

This problem with the trapezoidal profile, which is the most common method in many existing industrial controllers, could be solved by modifying the superposition scheme. One possible solution would be to reduce the velocity during the first motion after new target coordinates have been specified, and add a second motion with reduced speed in such a way that the combined speed of the whole motion remains equal to the maximum permitted. Both motions are completed at the same time. The second motion corrects the first motion in order to arrive at the new target. This assumes that the permitted acceleration is large and acceleration time is negligible. The resulting velocity profile is shown in Figure 20. The dotted line represents the main movement, the dashed line is the corrective motion units, and the solid line stands for the motion profile of the resultant movement. Knowledge of the manipulator velocity and position is used to calculate the velocity and duration of the corrective movement without using the complicated optimization algorithm. This type of trajectory replanning requires relatively few commands to be issued to the controller and does not overload the system.

2.2.2. Animated Simulation and Analysis of Robotic Systems

A number of graphic simulators have been developed to design and analyze robotic systems (Klein and Maciejewski, 1991). Since the design of a robotic workcell is very complex due to the many closely related design parameters and many commercially available

Figure 20. Superposition algorithm for a trapezoidal velocity profile: correction of the main motion with the additional corrective one.

robots that can be selected, optimal robot selection, design and placement is an extremely difficult and time consuming task. The existing robotic simulator packages allow the engineer to examine alternative robot models from a library for integration into a workcell, considering different machine layouts, material and tool flow and control strategies (Nof, 1991a,b). The engineer can select off-line the optimal placement of the robot in the workcell and ensure that the robot can reach all necessary points without collisions. With the increasing complexity of modern manufacturing systems the number of alternatives to be analyzed during the planning process has grown. The use of graphic-based simulation allows to evaluate many configurations in a timely manner and gain insight into the complex interaction between different system components before purchasing and installing expensive equipment.

The form of graphic display translates an enormous amount of numbers into an easily interpreted visual display. Instead of numerical analysis which outputs an overflow of meaningless numbers, animated visual simulation provides a dynamic, realistic, three dimensional visualization of the mechanisms and therefore provides intuitive feel of problems and solutions. In addition, graphic simulators are important for planning trajectories and tasks. This includes generation and verification of trajectory plans, ensuring collision avoidance and optimizing motion trajectories. Graphic based simulators are excellent tools for programming and debugging robot motions which provides also for planning changes prior to implementation. New advances in robot ergonomics has been achieved regarding robot motions and workplace design by use of powerful graphic-based simulators. A robotic simulation package has been used to design an agricultural robot, plan its motions and verify its intelligent control system for a variety of unknown and changing environmental conditions (Edan, Engel, and Miles, 1993).

2.2.2.1 Animated Simulation and Analysis of Industrial Robotic Tasks

Robot tasks frequently include repeated sequences of operations (Edan and Nof, 1992). To increase productivity, these operations should be optimized by planning efficient motions. In planning robot work, both the method and the robot can be specified to optimize

performance. Performance of a particular robot depends on its design, its motions and the workcell setup. In addition, robot performance is highly dependent on the spatial geometry and distribution of the task points (Edan *et al.* 1991). Hence, specific robot designs can be selected for certain tasks and the robots motions and the workcell setup can be optimized to improve performance and reach productivity goals effectively. The various engineering techniques that can be used for this planning are described by Nof (1985), within the general context of robot ergonomics. In this context, ergonomic analysis of typical robot motions relative to specific types of tasks can lead to the development of general, systematic motion economy principles (MEPs). These MEPs are useful to improve engineering functions such as: robot selection, workplace design and motions planning.

In robot-MEPs, there are four main groups of principles (Nof 1991a):

1. robot motions
2. design of a robotic workcell
3. tooling and equipment and
4. sensors.

In this work, previous research in robot motion economy that was limited to laboratory or analytical experiments (Nof and Robinson, 1986) was continued and expanded using graphic simulation. Graphic simulators and emulators have been developed to design robotic systems by animated, graphical representation of the time-varying solutions (Klein and Maciejewski 1988; Nof 1991b). Through simulation, alternative manipulator models from a library can be examined for integration into a workcell, considering different locations of material, different operational methods of material flow and different motions of the robot. Predicted cycle times can be computed, and different scenarios can be evaluated not only for technical feasibility and functionality, but also to optimize plans and maximize utilization and efficiency. This work utilizes a graphic robot simulator to derive MEPS for a variety of robot classes in different group sizes.

Using graphic simulation, different representative robots were compared for their performance in a kitting task for different modes of motion through a component part bin for different bin positions. This type of task, which is common to many robot applications, was chosen because its cycle-time is determined directly by the robot motion and is relatively less dependent on extraneous factors. The general task for all the analyses was to move to a bin from a home position, then pick repeatedly several parts from the bin cells and return to the home position. Two different modes of motion through the bin were investigated. For generality, several bin positions were included with different locations and orientations of the bins. The simulations were performed on a Personal Iris workstation (Silicon Graphics) using IGRIP (Interactive Graphics Robot Instruction Program, Deneb Robotics). Robot graphic simulators/emulators such as ROBCAD, by Tecnomatix and IGRIP are three dimensional graphic systems for design, simulation, analysis and off-line programming of robotic systems. The simulator provides reachability evaluation and cycle time estimation. Robots can be selected from an existing extensive library of the most common industrial robots.

The four basic structure classes of robots that were selected: cylindrical, articulated, spherical and SACRA (selected compliance assembly robot). For each class, three different group sizes of robots were analyzed: small, medium and large. This set of robots was chosen to allow comparison between different classes of robots as well as within a specific class. The

group sizes were defined based on robots weight (taken from handbooks) and dimensions (taken from IGRIP or ROBCAD data). The small robots consisted of robots weighing approximately 100 lbs, the medium range 300 lbs and the large robots approximately 1000 lbs.

The type of task that was compared was kitting: the robot travels through a given set of bin cells, picking kit components into a magazine and finally depositing the whole kit at a drop point. For each cell, the robot travels to a point above the cell location, then moves into the bin cell, stops to pick up the part, moves back out to the same point above the bin cell location and continues to the next reachable bin cell. In order to increase productivity and minimize the travel time, the robots were programmed to move at the maximum possible velocity. Therefore, the velocity profile assumed was a trapezoidal. For short distances the velocity profile was triangular. Since the objective was to compare the effect of the main kinematic structure of the robots, the orientation of the end effector was not considered as a significant factor for this analysis and thus, not modeled. The robot was assumed to kit through all bin cells in the following two modes of travel through the bin cells.

A 4X4 bin of 16 cells was placed in front of each robot in three different orientations:

- A horizontal bin (B1: 0 degrees tilt);
- A diagonal bin (B2: 45 degrees tilt); and
- A vertical bin (B3: 90 degrees tilt).

Initially the small bin was defined with an individual cell size of 150 X 150 X 20 mm. The medium and large bins were scaled up respectively multiplying by a factor corresponding to the average dimensions of the robots in each group size. In addition to the orientation, for each bin three different heights and distances were taken for its location based on the maximum and minimum reachable height and distance for each robot. The three different heights and distances for the bin locations were calculated as follows:

$$\begin{aligned}
 H_1 &= 0.7 H_{\min} + 0.3 H_{\max} \\
 D_1 &= 0.7 D_{\min} + 0.3 D_{\max} \\
 H_2 &= 0.5 (H_{\min} + H_{\max}) \dots\dots\dots [13] \\
 D_2 &= 0.5 (D_{\min} + D_{\max}) \\
 H_3 &= 0.3 H_{\min} + 0.7 H_{\max} \\
 D_3 &= 0.3 D_{\min} + 0.7 D_{\max}
 \end{aligned}$$

As a result, each individual robot was analyzed with 27 different bin positions (orientations and location combinations) and each global robot class was analyzed with 81 different bin positions (orientation and location combinations). Overall performance was derived by accumulating the results of all bin positions for each robot type. Hence, each result is the average of all 81 bin positions and therefore, indicates overall performance. The effect of the different modes of motion was evaluated for each specific position separately by accumulating the results obtained for each distance, height, and orientation. Each result presented is an average of all tested cases, *i.e.*, for each orientation (B_1 , B_2 , B_3) the results are the average of the results obtained for all distances (D_1 , D_2 , D_3) and all heights (H_1 , H_2 , H_3). Thus, each result presents the average of 9 cases.

The criteria for comparing the robots performance were:

- **Reachability:** The number of cells that each particular robot succeeded in reaching in the bin (out of 16 possible bin cells).

- **Cycle times:** The total time it took the robot to accomplish the kitting task is divided by the number of reachable cells. Since the motion time depends on the tool center point (TCP) acceleration and since the different robots simulated had different TCP accelerations, these were normalized to equalize the conditions for comparison. The normalization was performed for each group size separately according to the robot's TCP accelerations as derived from the robot library.
- **Arm joint utilization (AJU) factor:** Arm joint utilization is calculated based on the total movement of all joints during the task performance. The arm joint utilization principle defined originally by Nof and Robinson(1986) is concerned with the design of tasks so as to make greater use of the robot arm joints whose motions require less effort of the drive system. This factor was calculated by using the link values at each position in the workcell as obtained by the graphic simulation language and calculating the difference between subsequent moves. For simplification, a unit move is considered equivalent for both 1 degree and 1 mm. To emphasis the analysis of the kinematic structure, only the first three degrees of freedoms were considered. A measure of drive system effort is generated by a weighted mean of the average AJU of each joint. The proportions used to generate the weighted mean are such that joint one's proportion is three times that of joint three, joint two is twice that of joint three and so on. Thus, the arm joint utilization factor (U_f) is expressed as:

$$U_f = \sum (0.5 \cdot d\theta_1 + 0.33 \cdot d\theta_2 + 0.17 \cdot d\theta_3) \dots\dots\dots [14]$$

where: $d\theta_i = \theta_{i,j} - \theta_{i,(j-1)}$, i = joint number, and j = position number.

By multiplying each mean absolute joint variation by its assigned proportional multiplier and summing up the resultant products, a comparative estimate is obtained of the effort required of the drive system performing the task in the given work position and of the wear of the joints.

These results indicate the following:

- For the 4X4 bin-cells' work areas, articulated robots had the best reachability, reaching the maximum number of cells for all locations and orientations, compared with the cylindrical, spherical and SACRA robots.
- Despite the large work volume of each robot, the work-area which is reachable for all available 16 bin-cells is very limited. Therefore, the selection of the bin position (location and orientation) has significant influence on the robots' performance.
- The SACRA robot performs the fastest overall (*i.e.*, for all bin positions); the cylindrical robot performs the slowest.
- For most bin positions, faster performance is achieved by widthwise travel through the bin for the articulated and SACRA robots; no significance of the mode of motion on cylindrical robot. For the spherical large robot, faster travel is achieved by lengthwise travel through the bin.
- The articulated robot has the minimum (best) joint utilization. When comparing specific bin positions, the cylindrical robot has the maximum (most wasteful) joint utilization. The

SACRA robot has the minimum overall joint utilization (for all bin positions) followed by the articulated robot.

- Vertical travel through the bin is less efficient than horizontal travel for the majority of bin positions for the small and medium articulated robot; for the medium and large spherical robot; and for the SACRA robot. The mode of motion through the bin does not influence the AJU of the cylindrical robot.
- Position of the bin (orientation and location) influences the reachability, cycle times and arm joint utilization. The significance of influence is robot type dependent.

2.2.2.2. Simulation of Agricultural Robots

To achieve fast and robust operation an intelligent controller capable of planning and executing tasks based on real-time sensory data was designed. This controller defined the organization of sensory data, the coordination and communication of information, the on-line integration of sensed data with motion planning and execution, the distribution of functionality among the computational subsystems and the coordination of interactions between these subsystems.

The data flow and plan execution had to be verified before implementing it on actual hardware to confirm that the control architecture performed effectively and ensure that algorithms and computer programs execute as intended. Since the intelligent control architecture involved functional and logical flow of data, which were of qualitative nature, it could not be verified using numerical algorithms and thus, conventional simulation methods were not applicable 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. A major problem approached was how to incorporate environmental changes into the simulations developed to verify dynamic planning and execution.

However, another important factor in the overall performance of an agricultural robot is selecting the appropriate type of manipulator for a specific task and defining its motions. When selecting a manipulator, the question is not only which type to use, Cartesian, cylindrical, spherical or articulated, but how many and how should they be configured to operate in concert? Is it better for all manipulators to do the same tasks in parallel or is it better for some manipulators to act as servers? The performance of a robot depends on its dynamic characteristics. However, when designing a robot, motor speed can be selected. Since drive price is not proportional to the power required, multiple slow arms might be less expensive than one fast arm. Moreover, by using multiple arms the complexity of the task performed by each robot may be reduced, and this can reduce the complexity of the machine design. Since there are many closely related design parameters, optimal robot selection and design is an extremely difficult and time consuming task which is not yet automated (Rivin, 1988). The complex and loosely structured environment in which the agricultural robot acts

adds to the complexity of this design. The solution for similar problems in industry is to simulate robot performance using commercial software packages which provide animated, graphical representation of the time-varying solutions (Tseng, 1988). By using simulation the amount of experimental research is reduced and the best robot can be selected and evaluated prior to prototype testing (Tseng, 1988). For the design of an agricultural machine, simulation can help identify the effects of crop variability on system performance (Miles and Tsai, 1987). When evaluating different machine designs during the development process, performance differences can be due to changes in the machine and the crop or interactions between the two. Through simulation, crop conditions can be repeated and altered to compare system performance for different machine parameters.

The complex task of designing an agricultural robot for harvesting melons was initiated by animating and visually simulating the manipulator motions and the melon handling (Edan and Miles, 1993). By animating the entire workcell and materials flows, relative motions between system components were observed. Through simulation in full three dimensional, shaded surface, real-time visualization, problems were easily detected and solutions proposed, simulated and evaluated in an interactive process. By designing different configurations and comparing various system models, the overall performance was evaluated and an initial design concept outlined. The graphic-based simulation results provided guidelines to specific design concepts and horticultural practices which were then thoroughly analyzed to further improve performance.

Alternative intuitive designs were compared by graphic simulations performed on a Silicon Graphics, Personal Iris workstation using IGRIP (Interactive Graphics Robot Instruction Program, Deneb Robotics, 1988). The IGRIP software is a computer graphics package for robotic workcell layout, simulation and off-line programming. The three dimensional (3-D) shaded color visualization provided realism that helped confirm the validity of the simulation and provided insight into the systems problems. Predicted cycle times were computed, allowing the different scenarios to be evaluated not only for technical feasibility but for determining harvesting rates as well. The paths necessary to accomplish the desired harvesting operations were planned, programmed and evaluated. Realistic results were obtained by simulating on actual coordinates of melon data. This was achieved by inputting the melons into the workcell at actual coordinates as measured in the field. The exact melons locations were read from data files.

Since distribution of the fruits along the row affects the design and performance of an agricultural robot, field coordinates of cantaloupes, *cv. Superstar*, 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, 1993). These data served as input to all simulations performed. However, since there are different crop conditions that can be adapted, the machine's performance was also evaluated for a variety of practiced horticultural conditions to determine the one that yields the best results (Edan and Miles, 1993).

Two different types of commercially available arms obtained from the IGRIP library were simulated: a cylindrical robot (Seiko/RT300) and a Cartesian one (Amada/Optimo). All actuator speeds of both robots were assumed to be equal. The simulation results performed on fruit coordinates of 10 different plants indicated that the Cartesian robot was 4% faster than the cylindrical one (Edan and Miles, 1993). Based on these results, the Cartesian robot was selected as preferable. The average cycle time was 10.8 seconds per fruit. Since this speed

was considered to be slow, further simulations were conducted to improve performance (Edan, 1990). Two different melon fruit conveying methods were modeled and evaluated. The first consisted of a box with individual cells for fruit attached to the tractor (Figure 21). In this case, the robot arm must pick and convey each fruit. To reduce cycle time, the required motions for transferring melons were reduced by adding an intermediate conveyor that approaches the robot arm and transfers the picked fruit to the main conveyor (Figure 22). The robotic arm approaches the fruit, picks it up, transfers the picked fruit to the nearby intermediate conveyor, places the fruit and approaches the next one while the intermediate conveyor conveys the fruit to the main conveying system. The speed of the intermediate conveyor actuators were taken as equal to the robot's actuators. The main conveyor consists of two conveyors that index when a melon is placed on it. Both main conveyors are beside the bed, where they don't interfere with the robot's motions. Adding an intermediate conveyor that approaches the robot arm and transfers the picked fruit to a main conveyor reduced the average cycle time more than 30% to 6.2 seconds per fruit (Edan and Miles, 1993).

Figure 21. Animated simulation of melon harvester with box-type conveyor

Figure 22. Animated simulation of melon harvester with intermediate conveyor

These simulations served as the initial stage of the design process. By assessing alternative concepts and designs on-line, the outcome of new ideas were verified immediately. The graphic-based simulation indicated that a Cartesian robot is faster and one arm could not harvest at a reasonable picking rate. Based on these conclusions, an analysis to determine the optimal number of arms, the configuration (serial, parallel, tandem), actuator speeds and best-fit horticultural practice was conducted. However, since the final evaluation needed to be based on an extensive statistical analysis and included a multitude of design parameters, alternatives were enumerated by applying numerical simulation tools. By applying numerical, simulation tools the time consuming process of designing three dimensional graphical displays and programming the various CAD designs was avoided. Performance parameters such as average cycle times, percentage of successful cycles and utilization of manipulators involved were predicted and thereby the best number of arms, preferred configuration, actuator speeds and best horticultural practices were determined. The best harvest cycle time (1.31 seconds per fruit) was achieved for two tandem arms operating on a proposed horticultural practice that consisted of one plant on each half row in an alternating sequence (Edan and Miles, 1993). The final designed prototype (Benady *et al.*, 1991), shown in Figure 23 was very similar to the initial concept derived from the graphic simulation.

Since all computed cycle times are based on the actual robot motions, it was important to plan the motions before conducting the statistical analysis. Through graphic-based

simulation the exact motions of the robotic melon harvester were defined, programmed and evaluated. The robot arm cycles through the following steps:

- Move to a point located 25 cm vertically above the fruit to be picked;
- Move to fruit;
- Send "close" signal to gripper;
- Wait for signal from gripper that fruit has been grasped;
- Move up 25 cm;
- Send signal to detacher that OK to detach;
- Wait for signal from detacher that stem has been cut away;
- Move towards conveyor at a distance of 25 cm from final destination;
- Move to exact final destination; and
- Wait for signal from gripper that fruit has been released.

A sequence of motions with an intermediate conveyor is (Figure 24):

- The robot approaches the fruit, picks it up,
- Transfers the picked fruit to the nearby intermediate conveyor,
- Places the fruit on the conveyor, and
- Approaches the next fruit while the intermediate conveyor conveys the fruit to the main conveyor system.

Figure 23. Design of Field Prototype

The gripper's motions were programmed according to the actions of a field tested gripper (Wolf *et al.*, 1990). Input/output signals were added to synchronize between the gripping device, the detacher and the robot. The gripper cycles through the following steps as shown in Figure 25:

- Grasp the melon, both jaws are closed around the fruit;
- Wait until robot signals that melon has been lifted slightly from the ground;
- Signal to the detacher to activate the cutting devices;
- Wait for signal from detacher that stem has been cut;
- Signal to robot that melon has been detached from vine;
- Wait for signal from robot that has reached conveying system;
- Release melon, open jaws enclosing fruit; and
- Signal to robot that melon is released.

The cutter returns slowly but this happens during transfer to the conveyor which is part of the operation cycle of the robot arm.

Figure 24. Animated simulation steps with an intermediate conveyor

Figure 25. Animated simulation of the gripper cycle

2.2.2.3 Simulation of Intelligent Control Systems

The control system for the agricultural robot is based on the blackboard model (Edan *et al.*, 1993) since it allows flexible interaction with different sensors and actuators and provides a very powerful mechanism for structuring various types of sensor data and integrating diverse information. In general, sensor data fusion requires a consistent representation and combines qualitative methods (rule-base, heuristic) and/or quantitative methods (numerical solutions). In the agricultural environment this condition is enhanced since the domain is not well defined or structured and therefore requires methods for quantifying the qualitative defined domain. In the blackboard model, diverse types and sources of knowledge can be integrated and both quantitative and qualitative methods combined (Nii, 1987; Engelmores and Morgan, 1988). For each sub-problem and at each stage of the solution formation, the best knowledge representation and solution strategy can be selected. Opportunistic reasoning is applied, that is which knowledge module to apply is determined dynamically, one step at a time, rather than in a fixed and systematic sequence. This is especially important in the dynamic and ill-structured agricultural domain which requires application of different algorithms for different

environmental conditions. For example, on a sunny day one algorithm will be employed versus a different one which will be executed on cloudy conditions.

The functional architecture for the intelligent controller was implemented using IGRIP which provides a Graphic Simulation Language (GSL, 1988) that allows a logical program structure. GSL provides the ability to construct programs for individual devices in a simulation to govern their actions and behavior. Animated motions were displayed on the screen in three dimensions to simulate the actions commanded by the robot and gripper controllers using IGRIP.

At the first stage, the input to the simulation were data files containing randomly distributed coordinates of melons along the growing bed. Next, coordinates of melons were simulated by entering the predictable spatial distribution according to the planting conditions: 3-4 melons located 80-120 cm apart. To evaluate the performance of the control structure based on realistic input, data of field measured field exact x, y coordinates of melons along the row served as input to the next set of simulations.

2.2.2.4. Robotic melon harvesting model

The robotic melon harvester 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 batch of fruit. Then the robot arm approaches each fruit and picks the ripe ones. Once all ripe fruit in the batch are picked, the tractor advances to the next batch. In the continuous mode, the robot arm picks the fruit while the tractor is moving along the row. A vision system provides the information required to operate the robot and consists of a far-vision system and a near-vision system. The far-vision system is mounted ahead of the current position of the vehicle and robot and determines melon locations. A fan disturbs the foliage to expose the melons which maybe hidden by the canopy. The near-vision system is mounted on the robot end effector robot and provides local guidance of the arm towards the next fruit to be picked.

Each module of the intelligent controller was connected to a different device in the IGRIP workcell and had a separate GSL program. All programs were executed concurrently. The simulated modules are described in Figure 26. All data were placed on the blackboard. The blackboard was implemented by creating data files accessible by all modules. The integrity of the data accessed asynchronously by the different processes is maintained by providing transactional access to the shared database. Any reference to the blackboard is part of a transaction. Each transaction is maintained by a time flag to ensure consistency among interacting processes. The different modules keep track of their current status by indicating time flags in a file. Each module has a special dedicated control file containing a number that indicates its current status. Thus, all modules can check the performance of others and react accordingly. Before accessing data, each module checks its own control file and the control file of the module responsible for deriving the required data. Only then, the actual data file is accessed, ensuring that it is the correct one.

The far-vision system was simulated by a process that provides the fruits locations at distinct time intervals. These locations are taken from data files that contain measured x, y coordinates of fruit. Based on these locations, the controller plans the sequence of fruit to be picked and a path towards each ripe fruit. Very small melons are assumed to be unripe and thus, not picked. Once the robot arm is located above a fruit to be picked, it starts moving

towards it and is guided by the near-vision system which also determines its exact ripeness stage. The near-vision system is implemented by simulating a process that generates various random conditions to simulate dynamic conditions. 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 simulation process continues until all ripe fruit along the row have been picked. Various algorithms for advancing the tractor along the row, defining the sequence of fruit to be picked and guiding the robot towards the fruit were evaluated using graphic simulation (Edan, 1990; Edan *et al.*, 1993). These consisted of heuristics and optimization procedures for task planning, motion planning and execution in a changing environment. The heuristic planning algorithms were implemented by rules in a CLIPS (CLIPS, 1989) program which 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.

Figure 26. Structure of the simulated control modules

Verification of the proposed control architecture for the robotic melon harvester was accomplished by measuring the response to perturbations in input conditions. By varying the following different parameters the logical flow of the data was ensured for various scenarios:

- 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; and
- Detection certainty factors.

To produce a different test case for each simulation, number of ripe fruit and detection Certainty factors were varied by assigning to them random numbers (generated from a uniform distribution based on the computer clock). 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. By changing the threshold value, the percentage of ripe fruit was varied, thus simulating different conditions during the season (at the beginning of the season very few fruit are ripe and the percentage of ripe fruit increases as the season progresses until a peak value is reached; from there on the number of ripe fruit versus unripe decreases). 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 actual detection of fruit). Thus, the dynamic planning algorithms were evaluated for simulated real-time changes in the sensory data.

2.2.2.5. Simulation Results

The logical flow of the data was simulated through all stages: from its input from sensors through the transformation to information, planning of the tasks based on this information and modification of plans according to dynamic conditions. 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.

By animating and visually simulating the operations under varying input conditions and system parameters, programs were debugged and corrected on-line. Figure 27 demonstrates a case where an incorrect operation was detected: an unripe (dark-green) melon has been picked. This happened because the robot became ready for the next operation before the "near" vision

system had detected the ripeness stage of the fruit. Therefore, the robot read the available ripeness data on the blackboard, which related to the previous fruit which had already been picked and started picking the current fruit without waiting for a signal from the "near" vision system indicating that it had detected the ripeness stage of the current fruit under investigation. To eliminate the problem, a time stamp flag was added also to the "near" vision process to ensure that the robot refers to the fruit that is currently under investigation.

Through an interactive process of simulating and visualizing the motions, for varying input conditions and system parameters all algorithms were verified and it was ascertained that each sub-module 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 which might be damaged by glitches or bugs in the software. 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 verified.

Figure 27. Animated simulation reveals inability of sensor to detect ripeness before the robot harvests the melon

2.2.3. Basic Concepts of Designing a Selective Harvester for Muskmelons

The robotic harvester design parameters were based on results of simulating motions required to grasp, detach, and transfer the fruit, based on measured locations for field-grown melons (Edan and Miles, 1989, 1991; Edan, 1990). Additional design guidelines for the research prototype were:

1. Flexibility to be adapted to perform other tasks and with other crops,
2. Provide support for development of hardware and software, and
3. Provide throughputs of 700 fruit per hour or more.

Therefore, a heavy-duty mechanical and electrical system was designed. Economical aspects were given lesser priority at this stage although that will be a major factor to consider during commercialization of these technologies.

The motion control and image processing hardware were personal computer based because it meets constraints for portability, environmental conditions, real-time performance, and it costs less than other, dedicated hardware. A vision system provides information of the melon locations. An air blast disturbs the foliage and exposes melons which may be hidden by the canopy. The vision system consists of a far-vision sensor and a near-vision sensor. The far-vision sensor scans scenes ahead of the robot's current position to detect melon locations. The near-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.

Two different modes of vehicle advancement along the row were evaluated: step and continuous modes. In the step mode, the vehicle advances along the row and stops at each batch, or hill of fruit. While the vehicle is stopped, the robot arm approaches each melon and picks the ripe ones. Once all ripe fruit in the batch are picked, the vehicle advances to the next cluster as determined by the far vision system. In the continuous mode, the robot arm moves towards the fruit and then follows it while the vehicle is moving along the row. This requires close coordination between the robot motion controller and the near-vision sensor which provides rapid updates of vehicle speed and the relative position information between the gripper and the melon.

The first set of field experiments were conducted in the step-mode because:

1. This mode is simpler and by first testing all components in this mode we could concentrate on the fundamentals essential for proving the feasibility of robotic melon harvesting (vision and motion control: ability to locate melons, reach and pick them), their integration and avoid dealing with the more complicated problems such as picking during motion.
2. There were several problems encountered in implementing multi-tasking on the PC, which was essential for the continuous mode operation (image processing had to be conducted simultaneously with motion control). The problems were due to unavailability of OS/2 drivers for both the motion control and image processing hardware and that the continuous mode operation required development of additional algorithms for dealing with dynamic tasks.

2.2.4 Overall System Design

The robotic melon harvester consists of a robot arm mounted on a mobile platform which is drawn by a tractor (Figure 28). The platform is a rectangular steel frame with two wheels at the rear and a hitch at the front attached to a drawbar. The platform is divided into two areas. The front area contains one of the visual sensors (far-vision) and provides a clear field of view of the whole bed for detection of oncoming melons. The Cartesian manipulator is located in the second area right behind the first one. The manipulator's workspace is 1.5 x 1.5 m. This enables access to any location of the melon across the bed (width 1.2 m) and to the special conveyers positioned along the platform on both sides. The vertical clearance of the gripper arm is 0.9 m. which is enough to pick a fruit and load it into the conveyers located along both sides of the platform. The gripper (Wolf *et al.*, 1990) is attached to the robot arm in a flexible joint to absorb side loads caused by horizontal motions during fruit picking. The manipulator is powered by electrical DC servo motors with encoder feedback. Electrical DC motors were preferred at this stage to ease interfacing to PC control hardware. The gripper is operated pneumatically.

Melons tend to cluster in batches close to the plant. The maximal planting distance is 1.2 m. Therefore, there could be two plants at a given time within the robot's workspace. In the step mode, this minimizes the number of stops the tractor needs to make along the row. In the continuous mode, this increases the chances of picking all fruit since a greater tolerance in the selection of the point of interception of fruits exist (Edan, 1990).

Figure 28. Design of the Robotic Harvester

The first, far-vision sensor is located on the aluminum frame placed on the platform. The field of view of the camera is approximately 1.0 x 1.0 m. from a height of 1.0 m. Blowers mounted on the platform disturb the foliage over the melons to expose them to the camera. The second, near-vision sensor is mounted in the picking area on the gripper itself at a constant height. It can be considered as an *eye on the hand* type that provides final information to direct the picking mechanism. The near-vision camera is located at a smaller height and has a narrow field of view.

The robot control unit consists of manipulator control and carrier control. The manipulator control components include a PC motion control interface and a custom designed robot control unit which was developed and manufactured by this project team. A fully autonomous controlled system would be optimal for the carrier, however this was not implemented in order to concentrate on research and development issues related to robotic melon harvesting. Nevertheless, the control unit was designed so that it could be easily expanded to include autonomous control. At the present stage, a carrier propelled by a hydraulic motor connected to a tractor was employed. The carrier is moved to the row by a tractor, and it is propelled along the bed by the hydraulic motor. This motor is computer-controlled through the robot control unit.

The robot control unit consists of:

1. Motion controller;
2. Amplifiers for motor control signals;
3. Switchboard for input and output signals;
4. Movement decoder;
5. Solid-state relays (optoelectronics);
6. Power supplies;
7. Sensors;
8. Emergency block.

2.2.4.1 Motion controller

A Galil DMC-630 PC servo control card was used to interface with the amplifiers. This interface card provides for fully programmable motion control of three servo motors with encoder feedback. The controller contains an extensive instruction set for executing complex motion programs. Instructions are sent to the DMC-630 via the PC Bus. A FIFO buffer enables fast pipelining of instructions. Programs can also be downloaded from the PC into the DMC memory. Modes of motion include independent or vector positioning, contouring, jogging and homing. The motion profiles for each motor can be specified separately or as a sequence of coordinated vectors. The coordinated mode provides linear and circular interpolation of two axes with continuous motion at the programmed levels for velocity and acceleration. Motion commands in a program can be executed using conditional statements which make decisions based on the logical state of I/O lines and motion parameters. Variables allow parameters to be input or modified during program execution. Error handling features include automatic shut-off for excessive position error, limit switch inputs, emergency stop inputs and programmable torque limits. Complete status reporting and positioning monitoring functions are included. The DMC-630 also contains 8 uncommitted input lines and 8 uncommitted output lines.

The complete DMC-630 servo system (Figure 29) includes the PC, DMC-630 motion controller, a motor, incremental encoder and power amplifier for each axis of motion, plus external switches such as end-of-travel and homing inputs. The ICB-960 interconnect board simplifies connections. Circuit schematics are shown in Appendix A.

- PC: Sends high-level commands to the DMC-630. Motion programs can be stored in the DMC-630 memory.
- DMC-630: Performs all the time-intensive functions of motor control, which includes generating motion profiles and position trajectories, decoding the encoder feedback and

comparing it with the programmed position, stabilizing the servo system, outputting a motor command to the power amplifier, and reporting errors and status of the system.

- **Motor:** Up to three DC motors may be connected to and controlled by the DMC-630.
- **Power amplifier:** Amplifies the DMC-630 command signal to the current necessary to drive the motor with the given load. For each axis a Galil unregulated DC power supply (PS-30/160) specially designed to complement the ESA servo amplifiers was used. The analog output from the DMC-630 to the amplifier varies between ± 10 volts. The DMC 630 provides a pulse-width-modulated output for switching transistors directly.
- **Encoder:** Translates motor motion into an electrical signal decoded by the DMC-630 as the motor position. Each axis requires feedback from an incremental encoder with two channels in quadrature, CH A and CH B (BEI - E203-256). The DMC-630 performs quadrature decoding of the encoder signals, resulting in a resolution of quadrature counts (4 X encoder cycles).
- **External inputs:** Include limit switches and emergency stop inputs that can be connected to the DMC-630 to prevent system damage. Other inputs and outputs are defined as necessary during field testing.

Figure 29. The DMC-630 Servos

2.2.4.2 Signal amplifiers.

Signal amplifiers execute the servomotors control that move the manipulator according to the software program. Dedicated Galil ESA 10/160 PWM servo amplifiers were used for each axis. The amplifier outputs are connected to the motors by power relays producing an operational current of up to 50 A. A current of 3 A is necessary for switching on the relay while, for maintaining it in the operational state 300 mA is sufficient. An automatic switching circuit is used for the relay power sources to avoid overheating the relays during prolonged operation. The high-power block that also controls the robot gripper is used to switch the relays on, while for maintaining them in the operational state, the small-power power block is used (the one that provides power to the robot's sensors).

2.2.4.3 Signals switchboard.

The following inputs and outputs were necessary:

Inputs:

- Gripper state sensors (3 bits)
 - ground 1
 - ground 2
 - status (open/close)
- Encoder for path traveled (8 bits input)
- Encoder for carrier turn angle (8 bits input)

Outputs:

- Gripper control (2 bits)
 - grip
 - cut
- Tractor motion (2 bits)
 - forward
 - backward

Since the DMC card provides only 8 inputs, it was necessary to design and construct a switchboard to connect additional signals. The switchboard is computer-controlled. The actual inputs to the DMC are multiplexed depending on signals received from the DMC outputs which were established according to Table 6.

Table 6. Truth table for switching multiple motion control inputs

FUNCTION	BIT NUMBER						DMC COMMAND
	B ₅	B ₄	B ₃	B ₂	B ₁	B ₀	
Sensors state	0	0	0	0	0	0	OP0
Grip	0	0	0	0	0	1	OP1
Cut	0	0	0	0	1	0	OP2
Path traveled per revolution	0	1	0	0	0	0	OP16
Number of revolutions	1	0	0	0	0	0	OP32
Tractor forward move	0	0	1	0	0	0	OP8
Tractor backward move	1	1	1	0	0	0	OP56
Stop tractor movement	0	0	0	1	0	0	OP4
Reset encoder	0	0	1	1	0	0	OP12

The signals switchboard is controlled by the computer through a decoding device. The switchboard connects the source selected according to the software (outputs set according to Table 6). The input port of the computer interfaces through the optoisolated, solid-state relays and controlled inverters block. The controlled inverters form the polarity (active high/active low) of each signal according to the exchange protocol.

2.2.4.4 Gripper state sensors.

To avoid melon damage and possible mechanical damage to the gripper at the moment of the stem cutting in case of an incomplete grip, the gripper is equipped with an "open/shut" sensor which informs the computer of the gripper's readiness for the stem cut action. Additional sensors (ZH and ZL) are used to protect the gripper from crashing into the ground and to signal that the uppermost point on the Z axis has been reached.

2.2.4.5 Path traveled sensor.

The path traveled by the carrier is determined by decoding signals of an incremental optical encoder Renco (model R-250) which is attached to the tractor's wheel. The encoder produces 256 pulses per revolution. Resolution was increased to 1024 pulses using the Full-Quadrature Decoding Method (FQD). The underlying principle of FQD is, that since the outputs of the encoder change state four times during a single quadrature cycle this information may be used to obtain four times the resolution over the Quarter-Quadrature Decoding (QQD) that for a long time was the most popular decoding method. The optimal resolution can be selected by dividing the pulses. The resolution with 256 pulses per revolution is 0.92258 pulses/cm. The output of the decoding scheme is a byte which has the value equal to the counted pulses since last decoder reset. The reset is used for setting the counter's value to zero.

At the first stage of development, the path traveled sensor was connected to the DMC through the aforementioned switchboard. This was sufficient since the encoder signals were fed to the computer only after the carrier came to a full stop. However, for picking in the continuous-mode, it was necessary to continuously access the encoder's data in order to compute in real time the relative position between the axis and the melons. This means the switching method could not be used in the continuous mode. Therefore, the converted encoder data were connected through an optoisolated, solid-state relay into the computer via the computer's standard parallel interface. This also enabled parallel reading of both encoders for future developments. The PC has 8 input bits available: five most significant bits from address 0x279 and the three least significant bits from address 0x27A_H. Bits 2-6 from the resultant byte are inverted by port hardware and hence, it was necessary to invert them in software. The desired position value is derived from these two addresses by logical AND, OR, and NOT operations.

2.2.4.6 Carrier turn angle sensor.

To determine the turn angle, an encoder is envisaged whose signals will be decoded in the same way as those of the encoder of the path traveled, as described above.

2.2.4.7 Movement decoder.

Although at the current stage there were sufficient outputs supplied directly from the DMC, the system was designed so that future developments could be easily integrated.

Therefore, the outputs were decoded through a 74191 decoder (*i.e.*, 2 actual outputs from the DMC enables selection of 4 control outputs). Not all outputs of the decoder were employed. The movement decoder is a device that, besides the decoding operation, is equipped with a control signals formation circuit whose signals control the movement of the robot's carrier according to the program.

2.2.4.8 Optoisolated, solid state relays block.

To prevent damage to the computer and to eliminate false signals due to noise, an optoelectronic converters block (OCB) is used with optoisolated, solid-state relays. The OCB includes INPUT/OUTPUT optocouplers (Gordos IDC5 and Gordos ODC5), that provide an optically isolated interface between computer and external devices. INPUT modules are used to monitor the status of the sensors, while the OUTPUT ones make it possible to drive the manipulator's solenoids for gripping a melon and cutting its stem in accordance with the program.

2.2.4.9 Power supply blocks.

The control unit is activated by four power supply blocks. A Galil TO170 135V transformer is used for the high-power block, for amplifying the motor control signals. A high-power, 24-volt block supplies voltage to the robot gripper. Two low-power blocks (12 V and 5 V) are used for the sensors and encoders. Control circuits make it possible to switch power for every servomotor separately or simultaneously.

2.2.4.10 Sensors block.

The sensors block includes the sensors of the robot manipulator position, the additional sensors informing of the uppermost and undermost position on the Z axis and gripper state sensors. To prevent over running in case of program failure, inductive proximity sensors (Syrelec IACMN1202E2) were installed at both ends of all three axes to limit travel before the arm reaches the physical end of the track and can be damaged. An additional proximity sensor was placed at the home position of each axis. Three limit switches were placed on each axis: X (HX,RX,FX); Y (HY,RY,FY); and Z (HZ,RZ,FZ), where H is HOME, R is REVERSE, and F is FORWARD. The sensors are connected to the computer by an optoelectronic converters block. They make it possible to continuously monitor the manipulator's position. These limit switches are connected directly to limit switch inputs on the DMC. All other inputs were connected through the signals switchboard.

2.2.4.11 Emergency block.

The emergency block immediately turns off the servomotors if manual switches strategically located on the vehicle are activated or an emergency abort signal is received from a radio controlled unit. A Digital receiver, ALERT, operates at 80-140 KHz and is connected to the abort signal on the DMC-600 interface.

2.2.4.12 Electrical power sources

Electrical power was provided by two Honda generators: an ET4500 which supplies 4.5 KVA and an E2500E which supplies 3 KVA. During early field tests, several power failures (instability and unexpected power downs) caused mechanical and electronic damage to electro-

mechanical circuits. Hence, it was necessary to add an Uninterruptable-Power-Supply (UPS) (Micro UPS-110) to provide power during generator failures, and to guard the equipment from damage. A UPS stores electric energy in batteries, which enables it to continuous supply power even when the incoming power is disrupted. This system proved reliable and no power-related failures were encountered after its installation.

2.2.4.13 Cooling systems

Laboratory testing revealed natural ventilation was insufficient to keep the electrical components cool. Overheating was detected by measuring on-board temperature using a datalogger and thermocouples. Fans were installed at critical sites to exhaust the heat created by the electrical components and to provide stable temperatures within the operating range specified for the equipment.

2.2.4.14 Tractor control

The tractor which normally pulls the robot platform is moved by a hydraulic motor connected to the power-take-off shaft. The hydraulic motor provides a constant speed during the movement of the whole system in the field. The motor has capability to move the robot forward and backwards. The direction is determined by a hydraulic valve which has a control input line connected to the output port of the DMC controller. This provides control of the hydraulic motor by the control software which issues motion commands: Forward, Backward and Stop.

2.2.5 Software Design

The software was designed to meet the following two criteria:

- To provide a flexible mechanism for communication to the DMC600 from the C programming language; and
- To build on top of this interface a set of functions for commanding a 3 degree of freedom robot.

To achieve these design objectives, the software was given a hierarchical structure as illustrated in Figure 30.

Figure 30. Hierarchical organization of the control software.

This structure was chosen to relieve the casual user from the onerous details involved in commanding a robot motion and to allow to link the robot motions with other activities such as vision, sensing and control through a high level language. At the same time, however, it has been recognized that it is often the case that new capabilities must be added to the robot without a substantial expenditure of time or effort. To this end, a set of relatively powerful generic commands has been built for providing and fairly painless interface between what is seen at the user level and what takes place at the communications level. In general incorporation of new capabilities should require only that a new Interface command be developed at the C program level. It is also likely that the software at the DMC600 level will need to be modified or added to. Typically, this modification will only involve the recognition of a new piece of data or a program to exploit a previously unused capability of the DMC600. Much of the intermediate processing should remain unchanged.

As alluded to above, only those commands which fall into the category of "Interface Commands" would appear in a typical program. As a general rule, these commands are the absolute minimum required for controlling the robot such as performing initialization, calibration, and any appropriate robot motion or inquiries. A typical example of this type of function would be "ax()" which commands the absolute position of the x axis.

The set of functions grouped into the "Generic" Command category consists of those procedures which are predominantly responsible for interpreting the desired robot action and initiating some sequence of actions at the DMC600 level. As an example, the function "DMCquery" provides a general mechanism for requesting information from the robot. In order to accomplish this, it must generate an appropriate request for the DMC600, communicate with the robot and then parse the response into vector as appropriate.

Also grouped within this category are those commands which provide the final stage of processing at the level of the C program prior to the communication with the robot. Typical functions of this type are "DMCrun_prog()" and "DMCset_var()" which are, respectively, responsible for initiating the execution of a program or for setting a variable at the DMC600 level.

The final set of C functions is responsible for the actual communication with the robot. The most important function within this category is "DMCinterface()", the c function which accepts as an input a string which is to be sent to the DMC600 and returns the response of the control board.

Attention was given to the decisions which were made regarding the use of this software for controlling a robot and to the requirements for developing a program for controlling the robot. As a general rule, the software was designed to operate in a manner similar to the VAL programming language. Obviously, the requirements inherent in commanding a six degree of freedom articulated manipulator are considerably more demanding than anything which might be needed for a comparatively simple three degree of freedom Cartesian manipulator. However, the choice of VAL as a programming model resolved two fairly important decisions regarding the robotic interface: how to synchronize the actions of the robot with the top level program and what representation to employ for communicating a motion to the robot.

For reasons of consistency, the software was designed in such a way as to always command the robot to perform an absolute motion at the DMC600 level. Clearly by maintaining a uniform representation for the robot's position, the software required for

performing a motion at the DMC600 level is greatly reduced. In particular there is no need for either replicating what would essentially be the same code or for adding function parameters for determining whether a relative or absolute motion is to be performed. This decision is analogous to using the set of joint angles as the end result of processing in a standard robotic language.

As far as the synchronization of the robot with the top-level program, it is both more intuitive and safer to have the execution of the top level program be synchronized with the execution of a robot motion. As a practical reason, software developers have established that it tends, in general, to be simpler to debug robot programs which are synchronized. For all of these reasons, the vast majority of the languages currently being employed to control industrial robots follow this convention, hence, it make a great deal of sense to follow it here as well.

The "units" in which C commands are issued are user definable via the robot calibration procedure. All variables supplied as function parameters or returned as function values are of type double. Tables 7 through 11 provide a synopsis of the command language.

Table 7. Motion Commands

COMMAND	FUNCTION
ax(x)	Move to the absolute position "x" in the x-axis while maintaining a constant position in all other axes.
ay(y)	Move to the absolute position "y" in the y-axis while maintaining a constant position in all other axes.
az(z)	Move to the absolute position "z" in the z-axis while maintaining a constant position in all other axes.
axyz(x,y,z)	Move to the absolute position given by the coordinates (x,y,z). The move which ensues will be coordinated, i.e. the robot will be controlled in such a way as to cause all of the axes to complete their motion simultaneously.
x(x)	Move "x" units along the x-axis while maintaining a constant position in all other axes.
dy(y)	Move "y" units along the y-axis while maintaining a constant position in all other axes.
dz(z)	Move "z" units along the z-axis while maintaining a constant position in all other axes.
dxyz(dx,dy,dz)	Move (dx,dy,dz) units from the current position using a coordinated move as described above.

Table 8. System Inquiry Commands

COMMAND	FUNCTION
qax(), qay(), qaz()	Returns the absolute position of the robot on the x, y, z axis.
qinp()	Returns the input port status
qtsx, qtsy, qtsz	Returns the status of the x, y, z axis
qax, qay, qaz	Returns the current acceleration setup
qsp _x , qsp _y , qsp _z	Returns the current speed setup
qac _x , qac _y , qac _z	
qvar	

Table 9. Velocity Profile Commands

COMMAND	FUNCTION
vel(v)	Set the robot speed to be employed while performing a motion. The parameter "v" is intended to be used as a percentage of the MAX_SPEED which the robot should ever attain.
mag_acc(a)	Set the magnitude of the acceleration vector to be employed while performing the motion. The parameter "a" is a percentage of the MAX_ACC which the robot should ever attain.

Table 10. Other Control and Utility Commands

COMMAND	FUNCTION
home()	Send all axis to home position
stopall()	Immediate stop of all motions (via software).
set_outp	Set output port to desired value
DMCinit()	Initializes robot data structures and DMC600 controller.
DMCcal_robot()	Initializes calibration constants for robot. Requires that some mechanism be developed for moving the robot a measurable distance in each axis while recording the number of quadrature counts needed for performing this motion. At the time of this writing the hardware needed to perform such an operation had not been decided upon.
lin2quad()	Convert from linear units to quadrature counts
quad2lin()	Convert from quadrature counts to linear units
check_in_range	Ensures given coordinate is reachable; within robot dimensions
DMCset_velocity()	Calculate the velocity components for achieving a joint coordinated motion.
DMCmotion()	Set variables required for performing a desired motion and initiate motion.
DMCset_var()	Set a variable at the DMC600 level.
DMCrun_prog()	Run a program at the DMC600 level.
DMCpick_loop()	Run the pick-fruit program at the DMC600 level
DMCinit_robot	Init all robot parameters (speed, acceleration, gain)
DMCquantize	
DMCcalc_comp	
DMCset_velocity	
DMCquery	
DMCparse_response	
mainmenu, printmenu, motionmenu, setmenu, fruitmenu	Menu functions
printpos, printspeed, printaccel, printstatus, printinputs	Print functions

Table 11. DMC600 Communication Functions

COMMAND	FUNCTION
DMCinit_structure	DMCclear_buffer
DMCmailbox_check	DMCclear_incoming_characters
DMC_character_available	DMCresponse_character
DMCsubmit_string	character_position_in
wait_for_colon	wait_for_question_mark
DMCsubmit_command	DMCinterface

2.2.6. Software and Hardware Integration

2.2.6.1 Preliminary Integration

In Israel, software was written both at the DMC level and C level for testing all system components: each axis, inputs, outputs, limit switches, abort, *etc.* First, all software was fully tested on the three motors that were activated in the lab, not mounted on the robot. This included implementing the fruit-pick cycle (as derived from the animated, simulations performed at Purdue during the previous stage of the project) and validating its performance. During this stage, the basic software was modified and several additions were incorporated. This included increasing the efficiency of communication with the DMC-600 board, developing an efficient pick-cycle command at the DMC level (which included activating gripper with interaction with sensors), incorporating the necessary safety commands (limit switches and routines, abort routines); developing general menus for: motion commands (relative and absolute motions of each axis independently and combined motions, home motion); status commands (checking inputs, outputs, status registers); picking fruit (single/multiple coordinates obtained from vision, files, on-line)

Only after the software was fully validated in the lab, all motors were mounted onto the frame and initial validation steps were conducted on the full system. All units were first tested electronically, then at the DMC level and finally at the C level.

Calibration tests were performed to calibrate each axis (*i.e.*, define how many encoder units were equivalent to 1 cm) and to determine the systems accuracy and repeatability. Extensive testing was conducted on the robot to ensure reliability and consistency of both software and hardware. This included running the pick-cycle in an infinite loop several times on different combinations of fruit coordinates, activating each axis back and forth multiple times for different speeds, accelerations and positions, *etc.*

The next step was linking the robot programs with the vision programs. This stage was very time consuming due to several technical problems that arose during the linking process. What seemed to be the most trivial and simple step, took the majority of time to complete. However, this is frequently the case with integration.

2.2.6.2 Step-mode software

When in the step mode, the vehicle stops near every melon or group of melons and performs the picking loop without the vehicle moving. Most of the operations are closely monitored by the human operator as a cautionary measure. Consider the tasks required to pick the ripe fruit from a group of closely located melons as an example of the step mode. The first

step performed by the control software is initialization of the system. This step is similar to the initialization of the system in the on-line (continuous) mode and includes downloading some utilities to the Matrox vision board and DMC motion controller. Obviously, this operation is performed just once and is not included in each pick loop. After initialization, the far-camera is activated. The program begins an infinite loop and takes sequences of pictures of the whole melon bed. During this operation, the operator can change parameters of the recognition algorithm to obtain a best object detection. These parameters include threshold value which depends on the light intensity, and average melon diameter. Once the best possible detection is obtained, the operator interrupts the vision loop and coordinates of the objects from the last picture are stored by the system. Then the operator issues the move command to the program which resets the distance encoder to zero and sends the movement instruction to the controller which activates the tractor motor. The tractor movement continues until the operator interrupts it by a special command. The decision when to stop the tractor is made by the operator who stops the movement when the melons appear in the robot's workspace.

When the tractor movement is completed, the program reads data from the distance encoder and using the calibration coefficient computes the distance the vehicle moved. Using this distance and melon coordinates obtained by the far-camera, the control program can predict the approximate position of the melons in the robot workspace. The exact position can not be calculated because of substantial error in the distance determination.

The next step in the pick loop is sending the robot arm with the near camera mounted on it to the estimated melon position. The position of the arm is chosen so that the near-camera is placed exactly above the predicted melon position. The near-camera functions in the same mode as the far-camera. The operator controls the image processing performed by the near-camera and is able to change its vision parameters. When all the melons located in the view of the near camera are found, the operator interrupts the detection loop. The melon coordinates obtained from the last picture are used as final melon coordinates. Using these data, the program sends the robot arm to the position located exactly above the melon and starts the low-level pickup loop performed by the DMC controller. This pickup loop includes picking the melon, cutting the foliage and transporting the melon to the storage bin. This operation is repeated for every melon found by the near-camera.

When all melons found in the first batch are picked, the program tries to define which melons from the set of melons generated by far-camera are picked and which are not. To achieve this, the algorithm uses the principle of minimal error. For every melon found by the near-camera the distance to the predicted position for all melons found by the far camera is calculated. The melon from the far-camera set which implies minimal value for this distance is considered to correspond to the given melon. If the minimal distance is larger than an *a priori* fixed value (maximal possible position error), the melon is considered to be a new one, not detected by the far-camera. If after the first detection-picking loop there are unpicked melons, the program sends the robot arm to the first melon from this group and repeats the detection-picking operation described above. When all fruits found by the far-camera are picked, the control program activates the far-camera and the pick loop is repeated.

The step mode is a very primitive operational mode which does not require any parallel processing. Most of the operations are controlled by the operator, not automatically. Therefore, the control software for the step mode functioning was implemented under MS-DOS. Robot testing in the step mode was performed both in the laboratory and in a melon

field. In the lab, the goal of the robot was to pick small yellow balls which were used to imitate melons. The conditions for the experiment in this case were ideal: a flat floor inducing very small error in the distance measurement, clear field of view providing perfect object recognition, *etc.* In the lab experiments, which included testing of approximately 100 "melons", 100 % of the "melons" were detected and picked.

2.2.6.3 Continuous-mode software

The continuous-mode control software consist of two levels.

- **Low-level** - DMC software which is similar to the VAL language and used for the low level manipulator control by the DMC controller, and
- **High-level** - main control program, programmed in the C programming language and developed under MicroSoft C (ver. 6.00).

To enable full multitasking capability it is desirable to function under the OS/2 operating system which meets the multi-processing requirements. The control program was designed using multi-process technology where several processes run simultaneously and independently, and communicate to each other using different systems providing communication mechanisms: shared memory, queues, semaphores *etc.* One can see several independent threads (sub-processes), each having its own destination and controlling its own part of the system.

User interface thread (UIT): The user interface thread is the main process in the system. This is the first process run by the operating system. The UIT performs initialization of the whole system, loads configuration parameters from the special configuration file, launches all other threads and stops until user input is received. The user input is transmitted to the other threads by UIT via shared memory. Every input depended process has its own control variables which are shared and can be changed by UIT. The UIT also terminates all other processes in the system when the program is exiting.

Matrox thread (MT): The Matrox thread interfaces with the Matrox visual system and controls image processing performed in it. Depending on the control variables, the MT takes the next picture of the melon bed either from near-camera or from far-camera and updates *Melon queue* which is an array of the melons coordinates in 2-D space. This array is built in such a way that the first melon in this array is the nearest one to the gripper. Since the robot is moving, the MT should not mix already found melons with the new ones which is coming to the cameras field of view. The way MT does it, will be described later.

Robot control thread (RCT): The robot control thread performs control of the gripper movements and trajectory planning. The RCT monitors the melon queue and is waiting for the melon to appear in the gripper's workspace. When the melon comes close enough to the gripper, the RCT generates the optimal trajectory and starts *pick loop* - a special program for gripping of the melon. While doing this, the RCT uses NEAR camera for correcting the melon coordinates.

Robot interface thread (RIT): The purpose of the robot interface thread is to transmit the commands from different threads to the robot controller. The problem is that if two different threads try to pass the command to the controller simultaneously, the controller gets confused and generates an error signal. Therefore, the RIT opens the input queue available for all other processes and they write their commands asynchronously to this queue. The RIT itself transmits the commands to the controller sequentially.

Encoder thread (ET): The only purpose of encoder thread is to maintain the shared variable *Current velocity*. ET uses for this a special distance encoder mounted on the robot's wheel. The encoder value is requested every control time interval and according to the change of its value the robot velocity is determined.

Before the control program is started some initialization is required. These steps include loading of the image shell into the Matrox and loading of some low-level programs for the robot controller. The image shell is an internal Matrox program which controls internal Matrox operations. The program for robot controller contains subsidiary subroutines which help the main program control the gripper. These two loadings are done by special utilities from the OS/2 operating system. The main program is started after the Matrox and motion controller are ready to work. The UIT performs some necessary operations and launches all other threads. The next operation performed is the command to the tractor to move. This command is passed to the controller by RIT. After the command was issued, the tractor begins to move along the melon bed. Since movement has begun, the ET estimates current tractor (robot) velocity and initializes the *Current velocity* variable. From this point we suppose that velocity of the robot is known. During the movement, the MT permanently takes pictures from the FAR camera and creates queue of melons performing image processing of the received image. Since the platform is moving, the problem is not to mix melons from old images with the new one coming into camera field of view. MT tries to reach this by considering the distance which the robot moved since last picture was taken. The shared *Current velocity* variable is used for this purpose. While the melon is passing across the field of view of the far-camera, the system takes and process up to 8 pictures. These are used to locate the melons more accurately. If the melon is found by the image processing algorithm at least 5 times at the same place, it is considered a real object; otherwise, it is assumed to be a false detection (noise).

While the robot platform is moving, the first melon from the queue becomes closer to the gripper and when it reaches it workspace the RCT is activated. The first problem to be solved by RCT is prediction of *intersection point*. The intersection point is the point where both gripper and the melon could come at the same time. This point should be as close to the beginning of the work-frame as possible. If the point will be close to the end of the frame, it could be not enough time to finish picking. The location of the intersection point strictly depends on velocities of the robot and the gripper and from the current location of the gripper. Knowing this data, the RCT can calculate the intersection point. The calculation is performed according to the following simple formulas:

$$T_i = \frac{P_m - P_g}{V_g + V_t} \dots\dots\dots [15]$$

$$P_i = P_g + T_i \times V_g \dots\dots\dots [16]$$

The *Time of Intersection* (T_i) is calculated both for X and Y directions (for Y direction the tractor velocity is equal to zero) and the maximum time for either axis is taken for the calculation of the intersection point. After the intersection point is calculated, RCT issues the move command to the gripper. When the movement to the intersection point is about to be completed the RCT sends a request to the MT to activate the near-camera. The near-camera is

used in the system for correction of the predicted melon position. The prediction of the melon couldn't be computed precisely. It is caused by several factors: the velocity of the robot platform varies slightly and the field is not straight. These are reasons for considerably large position error which must be eliminated by near camera. The near-camera takes a picture at the time when it is located above the melon and returns the new melon's coordinate. From that moment, the system gets correct coordinates of the melon and starts superposition algorithm for replanning the current gripper movement. The current location of the melon plus a distance the robot is going to move while the gripper movement will be accomplished is taken as a final target for the superposition algorithm. This is also some kind of prediction and it obviously causes a position error, but since the distance the robot moves till it picks the fruit is very short, the error is very small and no additional corrections are needed.

Then the gripper starts a special program for picking the melon. This operation is controlled by a special low-level controller program. The only operation performed by the main PC program is sending the platform velocity to the controller. While gripping, the robot arm must not move relative to the melon. To achieve this, the robot arm moves with velocity equal to the platform's velocity in the opposite direction. Therefore, respectively to the melon the gripper keeps its place unchanged during the picking. After the picking is completed, the gripper closes around the melon, lifts it up from the ground, activates a special cutter and cuts the foliage and places the melon onto the conveyor. This operation finishes the pick loop for the current melon and the system starts to proceed the next melon from the queue (if there is a melon in the workspace).

2.3 Prototype Design

The performance criteria for the Cartesian robotic system were specified and several manufacturers of robots and industrial motion control systems were asked to supply the Cartesian robot system. No off-the-shelf system was capable of fulfilling the performance requirements for the robotic harvester in terms of payload, speed, flexibility in programming the control system, and incorporating real-time sensory-based trajectory planning. An order for a customized gantry robot was placed with Monforte Robotics, Inc., 30 Ward Avenue Extension, Trenton, New Jersey, 08619. The robotic manipulator was made available for acceptance tests at the manufacturing plant by members of the research team both from Purdue and Volcani. The system did not meet specifications, and after futile attempts by the manufacturer to rectify problems, the product was rejected.

A decision was made to design and construct the entire motion control system from industrially-available, OEM components. Selection of the mechanical and electrical components for the robot was made jointly by the Purdue and Volcani research teams at Purdue University, and continued via frequent Fax and electronic mail communications between the researchers. Sliding rails, motors, power supplies, digital control hardware components were selected, and were purchased by Purdue. At the same time the final mechanical design of the system and the dimensional drawings were completed at Volcani.

The manipulators are powered by electrical DC servo motors with encoder feedback. Electrical motors were identified as being the most appropriate for the prototype robotic harvester to ease interfacing to the PC control hardware, but this choice will not necessarily be the best for a commercial agricultural harvester once development issues are resolved.

After the research team at the Volcani Center completed the design of the Cartesian-robotic-manipulator, a decision was reached to build the entire robot at the site of design rather than at Purdue. Originally, plans called for Purdue to build an initial prototype and test the concepts in lab and preliminary field studies, and for Volcani to build a field prototype. However, this was not implemented because of budgetary constraints. Instead, the one prototype was constructed and tested at the site of the design: Volcani. This saved much of the communication, prevented errors and most of the shipping expenses required for an assembled unit. Thus all the components purchased by Purdue, such as sliding rails, motors, power supplies and digital control hardware, were transferred to Israel. However the mobile frame constructed at Purdue was not sent because the shipping costs were higher constructing a new frame at Bet Dagan. The prototype frame is shown in Figure 31 and is described below.

2.3.1. Construction of the Cartesian Manipulators

The function of the Cartesian manipulator is to move the end-effector grippers to the requested position along the X, Y and Z axes, and to lift the melon located by the machine vision system and deliver it to a conveyor belt. The grippers are mounted at the lower end of a vertical column, moving up and down in a housing (Z axis). This housing slides along a bridge, providing the travel across the frame (Y axis). The bridge travels along the frame on rollers mounted to its ends (X axis).

Figure 31. The Prototype Frame

The manipulators can be divided into the following elements:

- Frame members;
- Slide systems;
- Drive elements;
- Position and track limit controls;
- Protective elements.
- Power and control links (trailing and fixed).

2.3.1.1 Frame Members

All the frame members are made of high quality (6061) aluminum alloy. The X-axis consists of a stationary support frame and a traveling bridge. Appendix C contains mechanical drawings of the frame. The stationary frame is made of a pair of 19 x 165 mm plates, each mounted vertically to 12.7 x 100 mm base plates, all measuring 2030 mm in length. These present two sides of a nearly square frame, held together by two rectangular profiles of 8 x 80 x 80 mm and 2056 mm in length. A slide rail is mounted to the inside of each of the vertical plates. The traveling bridge consists of a rectangular frame, measuring 278 x 1980 mm on the outside, made of 19 x 125 mm plates, assembled by bolts and guide pins. Slide rollers are

mounted to the outside of the cross plates, to travel on the stationary rails, while an additional pair of rails is mounted on the inside of the bridge plates.

The Y-axis housing consists of a nearly square construction 160 x 172 mm (outside dimensions), made of two 9.5 mm plates bolted to two 12.7 mm plates. Four slide rollers are mounted to the outside of each of two opposing plates, to ride on the slide rails of the Y-axis bridge. The other pair of plates has slide rollers mounted to the inside of the housing to provide for the travel Z-axis column.

The Z-axis housing is made of two 12.7 X 80 X 1180 mm plates, bolted together to 3 spacer plates, to form an 80 mm square column. A slide rail is mounted along the outside of each of the two plates, to provide for the Z-axis travel inside the Y-axis housing.

2.3.1.2. Slide systems

Double Vee shaped, hardened and ground slide rails with matching grooved journal rollers were used to provide for smooth and precise travel of the manipulator parts. Made by Hepco (Middlesex, England), medium size slide ways (NM-44) were chosen, having 44 mm wide rails and 34 mm O.D. rollers. Factory-made carriage plates were mounted in place as complete units with rollers and other hardware. Each plate has two fixed-position rollers and two with eccentrically-adjustable shafts. Carriage plates with roller spacing of 150 mm were chosen to increase rigidity. All the rollers were protected by "Capseal" housings, filled with grease and provided with felt wiper blades to lubricate the slide rails and to inhibit rusting. Slide rails 1680 mm long were used for the X and the Y axes which provided a travel distance of about 1.4 meters, while rails 1140 mm long were used on the Z-axis.

2.3.1.3. Drive elements

The manipulator is actuated by three computer controlled servo-motors, one for each axis. The servo-motors (made by O.S.E.M., London, England) with on shaft mounted encoders were recommended by the manufacturer of the motion control system (Galil Motion Control Inc. Sunnyvale, CA), according to the load estimates and service requirements of the robotic melon harvester. The drive-trains vary slightly for each axis.

- **X-axis bridge drive:** The X bridge is actuated by means of two number 35 (3/8") roller chains, one at each end of the approximately 2 meter long bridge. The chain runs over drive sprockets at one end of the support frame and over idler tensioner sprockets at the other end. The chain-loops are closed with their ends anchored to plates on opposite sides of the bridge. Sprockets with 26 teeth and a pitch diameter of ~80 mm were used. The two drive sprockets are mounted on a single shaft stretching across the support frame, in order to keep the bridge aligned. The shaft 2150 mm in length is supported by two 20 mm diameter flange bearings mounted to the frame plates. The shaft extends 40 mm through one of the frame plates to accommodate an H.T.D. (High torque Drive) belt pulley. Torque is transmitted by means of an 8 mm pitch, 30 mm wide Gates H.T.D. power-grip belt which is a high traction, timing belt. The belt pulleys have 44 teeth and a pitch diameter of 112 mm. The servo-motor (SD40 R4-42, 106 mm diameter, and 260 mm long without encoder), with an in-line-mounted planetary 1:10 reduction gear drive (Bayside Controls USA, PG 90-10), and drive pulley are mounted via a tensioning plate, to the side of the support frame. This configuration of belt drive permits minimum overhang from the side of the support frame with only the belt pulleys extending outside,

while the 500 mm long motor and gear assembly are cantilevered over the inside of the frame.

- **Y-axis housing:** The Y-axis housing is actuated by an identical servo motor, reduction gear and power-grip belt and pulleys, however, only a single number. 35 roller chain was required because of the narrow spacing between the slide rails (212 mm compared to 2036 for the X bridge) and because of the smaller load to be moved. The X-axis servo must propel both the Y and Z axes, whereas the Y-axis only moves the frame for the Z-axis.
- **Z-axis column drive:** The Z-axis column is lowered and lifted by means of a cable and drum winch system. A 3 mm diameter stainless steel cable was attached to a thread grooved 80 mm diameter winch drum, in its middle (middle of cable and also middle of groove). The drum and the bearings of its drive shaft are mounted inside an extension of the Y housing. The cable ends are anchored to the top and bottom plates of Z column. One side of the cable is prewound on the drum, thus when the drum is rotated this side of the cable unwinds while the other winds on the drum pulling the Z column up or down respectively, depending on the direction of the turn. The drum is actuated by a servo motor (SD30F4-25, 106 mm diameter. and 197 mm long), through a 1:15 worm reduction gear. A motor spring counter-balance-mechanism attached to the shaft and Y housing, is used to reduce the load from the Z column weight by 5 to 8 kg, depending on the lowering rate of column, so that a smaller servo motor could be used. The counter balance spring together with the braking induced by the 1:15 worm gear prevents the column from slipping downwards when power to the servo motor is terminated.

The cable lift functioned well as long as the cable remained tightly wound around the drum. In a few instances the cable became slacked and slipped from the drum. In one incident, the Z column free fell to the ground. Therefore, to reduce the hazards to both people and equipment, the cable and drum lift were replaced by a chain and sprocket drive. A No. 35 chain (3/8") similar to those used for the X and Y drive was employed. A small housing containing a 17-tooth drive sprocket and two 11-tooth idler sprockets (to lead the chain around the drive sprocket) was mounted on the drive shaft instead of the cable drum. The relatively small sprockets were necessary because of limited space in the Y-axis housing. The 17 tooth drive sprocket has a 51.5 mm diameter compared with the 80 mm diameter of the cable drum. This increased the transmission ratio so that a higher motor speed was required for equal axial translation speed. However, it also resulted in an increased output torque which eliminated the need for a motor-spring counter balance mechanism.

2.3.1.4. Position and track limit control

The position of the robotic manipulator arm along its three axis is controlled by the computer, so that it can be sent to any desired point within the boundaries. However, to prevent over-running at the tracks ends in the event of program failure, proximity sensors were installed at both ends of all three axes to limit travel before the arm reaches the end of its track. These limit switches were connected directly to servo control hardware, bypassing software.

Three additional sensors - one for each axis, were installed close to the track limit sensors, to serve as a home position which is required for system initialization. In addition, mechanical stops were installed just beyond the track limit sensors at both ends of the X and Y axes and for the topmost limit of the Z axis. The down limit sensor of the Z axis functions

only when the gripper does not contact the ground, which would occur when operating over a deep depression in the soil (*i.e.*, a ditch).

However, for normal operation of the grippers and in order to prevent them from pressing into the ground, some means for sensing ground contact, was required. This was achieved by spring loading the top anchor of the cable which lifts and lowers the Z column. As long as the Z column descends freely, the upper section of the cable is not loaded (except for the spring). However when the grippers press against the soil or any other object such as a melon, this contact force is transmitted to the upper cable section which depresses the spring at the anchor point, activating the proximity sensor and limiting downward motion. After being stopped by the ground contact sensor, the controller is programmed to lift the Z column 30 mm to relieve the contact pressure and free the gripper to close around the melon. If the ground contact sensor fails, a backup sensor is activated by additional compression of the anchor spring, cutting power to the motor. A proximity sensor is mounted on the gripper open/close mechanism to sense complete closing of the gripper's jaws prior to operating the cutting device. In all, twelve proximity sensors were installed:

- Three for X-axis,
- Three for Y-axis,
- Five for the Z-axis, and
- One for the grippers.

2.3.1.5. Protective Elements

For field operation it was deemed necessary to guard against the grippers hitting rigid objects. Damage-avoidance was provided by mounting the gripper to the bottom plate of the Z column, via a spring-loaded four directional hinge. The hinge was designed to flex in all four directions of the X and Y travel, and to return to the normal position when the obstacle was cleared.

An electrical release disc brake (Warner Electric FERM 0.6 G-E560) was installed on the Z column drive shaft to prevent it from slipping downward when the servo motor power is off. The brake is released as long as power is "on" and actuates (brakes) when power is interrupted.

The slide rails and drive chains require shielding to prevent dust and sand from adhering to their lubricated surfaces. For the X-axis, sliding shields, made of 0.2 x 150 mm hard-rolled stainless steel, 1600 mm in length have been added. Four such shield sections are employed, two per side, attached vertically to the ends of the bridge plates. The shield sections slide in brass guides, installed along and across the X frame, and move in and out of position along the slide rails and the drive chains. Shields for the Y slide rails were made of a single ply polyurethane conveyor belt materials, 220 mm wide and 1800 in length. Two such shield sections are used, in a roll-up-shade configuration, running horizontally in guides on the top of the X bridge attached to each side of the Y housing. The shield belts are wound on or off spring-driven rollers mounted underneath both ends of the X bridge.

The shield for the Z column, is made from air sealed (silver color) nylon fabric sewn into sleeves. Two sleeves were employed, one for the section of column extending about the Y housing and the other for the lower part. The sleeves made of thin fabric, have an advantage over bellows because they fold to a very narrow height.

2.3.1.6. Umbilical cords for power and control

The X-axis drive motor and position sensors are mounted in fixed, non-traveling locations. The power and control lines for these items have been installed in a U-shaped, 80 mm wide aluminum duct bolted along the top of the X rail mounting plate to which the drive motor is attached. An identical duct had been mounted to the opposite plate. This accommodates a flexible "Nyltrac" link type cable and hose carrier. One end of the carrier is attached to the bottom of the duct, while the other end is doubled-up and attached to the Y drive motor, which is mounted on the X bridge. A second duct was mounted on the side of the X bridge. An additional carrier was fixed to the bottom of this duct close to the Y drive motor while the other end was doubled-up and attached to the Y housing to which the Z drive motor is mounted.

The cables for the Y drive motor and Y-axis position sensors were laid through the first carrier only, to trail along a single axis (X axis). The cables for the Z motor and sensors and the 8 mm air pressure hose for the actuating cylinders of the grippers run through the first and then through the second carrier too, in order to trail along two axis (X and Y), separately or simultaneously.

2.3.2. Design of the Carriage and Frame

Based on the conceptual design layout and dimensions, developed at Purdue University, a new mobile frame was built at the Volcani Center. This frame was designed to be used:

- To accommodate the robotic system for testing in the lab, and
- To serve as a mobile frame for extensive testing in the field.

With the progress of the research and development work, additional features were necessary for conducting lab and field tests. These have been designed and installed on the mobile frame. All the accessories, except for the robotic manipulator itself (which was bolted to the rear of the frame), were attached to the frame by removable clamps. This allows for easy removing, moving or adjusting when found necessary.

2.3.2.1. The Frame

The basic, mobile frame was constructed of hollow, rectangular section (RHS) steel, with 5 mm wall thickness. It measures 4520 mm in length and 2110 mm in width. The cross section of the longitudinal beam is 80 x 120 mm and the cross beams are 120 x 120 mm. The wheels were mounted underneath the rear beam having 7.50 x 16" implement tires, approximately 780 mm in diameter and 200 mm wide. Adjustment of wheel track from 1750 to 1950 mm was provided by shifting the wheel axles in or out, in their mounting clamps. The frame on its wheels stands 1010 mm high to the top surface. The frame is provided with two front legs to support and stabilize it in the lab and during trucking. When not in use, the legs can be raised and secured. The drawbar was made detachable (by two 3/4" bolts), in order to make the frame fit on an available 5 m. long truck for hauling purposes. The draw bar is provided with a screw-jack to adjust its height to the tractor hitch.

A secondary frame which accommodates the active vision equipment, was constructed above the front part of the basic frame. This superstructure measures 1960 mm in length and 2070 mm in width and is 720 mm high above the main frame. In order to provide for three direction positioning of the video cameras, a movable bar was clamped across the frame,

equipped with fasteners for two vertically adjustable camera tripods. (a third tripod was attached to the Y housing, to hold an additional camera for precision guiding of the manipulator arm.

A cabinet containing the motion controllers, a line to the 24 V transformer and other items, was mounted on the right side of the front cross beam. The cabinet, the dimensions of which are 800 mm height, 260 mm depth and 600 mm width, had to be mounted on a special stabilizing frame which was bolted to the front beam. All the power and control inlet and outlet cables were hooked to the cabinet by means of connectors (19 in all) installed in the side walls of the cabinet.

The Hewlett Packard Vectra 486 computer which serves as main executive controller of the robot, required a shock protected mount, to safeguard it during field operation. A special base was constructed to fit the dimensions of the computer, which were 210 x 520 mm depth and width and 610 mm height. This base was mounted via four shock absorber blocks to the left side of the front cross beam. The base was provided with buckles and straps, to secure the computer in place.

A special work station, for the computer programmer was provided in the front part of the mobile frame. The station consists of a seat for the operator and two stands for monitors, one with keyboard for programming and the other for the video camera image. The seat and the stands were attached to an additional cross beam made of 50 x 50 x 5 mm RHS, mounted about 400 mm beneath and 300 mm to the rear of the front cross beam. The seat, of the swivel type, with back and foot rests was positioned at one end of this beam. Each of the monitor stands swivel on the front end of a swing arm, which are adjustable in height and can be locked in position to suit the operator. The stands are provided with buckles and straps to secure the monitors.

A platform 450 mm wide and 1250 mm long was mounted crosswise behind the rear beam. The platform carries the generator which is the mobile power source for the robotic manipulator motors, controls, the computer, and air compressor. The latter was placed beside the generator on the rear platform, and connected via solenoid valves to the activating air cylinders of the grippers. Since the air compressor must be powered from the generator, a small one with a reservoir of 10 liters, a working pressure of 8 Bar, an output of 30 L/min, and powered by an 186 watt (0.25 H.P.) motor was chosen.

The mobile frame of the robotic melon harvester was designed to be drawn by a tractor. This presents problems such as:

- Controlling the forward travel or forward translation of the tractor in precise coordination with the robot controller; and
- Using a tractor with an internal combustion engine for laboratory research work.

To make laboratory research work possible, without to have to rely on a tractor, an electric drive unit was constructed and attached to the mobile frame. The drive unit consists of a fork with a 600 mm diameter wheel, mounted via a housing with a vertical pivot shaft (steering) to the center of the front beam instead of the drawbar. A 12 volt DC motor previously used to drive a tractor through the ground power take off (PTO), was slipped on the splined end of the drive wheel shaft. The drive unit has an output torque exceeding 500 Nm and an output speed of 3.5 RPM. providing 6 m/min travel speed for the mobile frame. The electric drive enables the computer to directly control the robot travel.

An encoder mounted on one of the rear wheels provides data on the forward travel of the mobile frame which can be coordinated with the precise requirements of the robotic manipulator. This drive unit can not be used in field tests because the single front wheel is centered on the frame and would travel on the melon bed. However the battery powered unit could be used to move the tractor through the field, if a tractor with ground PTO is used. This would permit the computer to control the tractor motion during field tests..

A battery-powered drive unit was mounted on the Power-Take-Off (PTO) shaft of a Massey Ferguson (M.F.) 35 tractor and used to propel the robotic frame during outdoor tests. Driving a tractor from an external source through the PTO shaft is possible if the tractor is provided with a ground-driven PTO option. This is available on some older model tractors, including M.F. and Fiat. When driven in this mode the transmission of the tractor is disengaged (neutral). However care must be taken to disengage the ground PTO prior to engaging the main transmission for normal driving. The battery-powered unit drove the tractor at about 90 meters per hour which was much too slow for field tests.

To drive the tractor at about 500 m/h, an hydraulic drive unit was assembled to operate from the hydraulic system of the tractor. The hydraulic unit consists of a high-torque, low speed (H.T.L.S) hydraulic motor (T.R.W Torque motor - MAE 24), a 12 Volt four-way solenoid directional-control valve, and a pressure compensated (P.C.) flow control valve.

The hydraulic-drive assembly was installed in series with the motor of the leaf blower which also operates from the hydraulic system of the tractor. A flow-control valve was mounted on a bypass between the inlet and outlet of the directional-control valve in order to adjust forward or backward travel speed of the tractor independently of the blower speed.

2.3.2.2. Foliage Fan

Since the fruit are often at least partially concealed by the leaf canopy, a turbulent air stream provided by a fan was used to expose the melons to the camera. At first, a 400 mm diameter fan in a duct 400 mm long was positioned on the side of the mobile frame. The air stream from the fan was directed to the melon bed in view of the camera. The fan was driven by a small hydraulic motor (Dowty H.M.30), at a speed of up to 2500 RPM. and was powered from the tractor. Although the fan supplied ample quantity of air in the lab, it did not do a satisfactorily job when tested in the field. An air stream coming from a single direction, uncovered some melons but blew leaves over others. The widely dispersed air flow was difficult to direct accurately so as to move the leaves without lifting the plastic mulch from the bed, or without creating a dust or sand storm.

A reciprocating air stream supplied from two opposite outlets and directed from both sides of the bed was constructed and studied. This was created by a high pressure fan with a directed air stream. An existing blower from a John Deere Cotton Harvester with a 400 mm diameter rotor was used. This was modified to provide twin outlets of 100 mm diameter. A butterfly type valve was fitted into each outlet, which were operated simultaneously and out of phase, by a crank mechanism, one opening and the other shutting the air outlet, alternately. The crank mechanism controlling the air valves was driven by a small 12 volt DC gear motor (car windshield wiper motor). After an initial trial in the field, the blower was mounted on the rear beam of the mobile frame, with outlets directed downwards. Air hoses 100 mm inside diameter and about 4 m. long were attached beneath the beams leading from the blower to air outlets mounted on the frame beside the camera. Direction and inclination of the air stream

are adjustable and so is the distance of the outlets from the center of the melon bed. The blower is driven by a small hydraulic motor (Dowty H.M.30) powered from the tractor. The long hydraulic lines were routed inside the frame beams.

When tested in a field with a highly developed leaf canopy, the alternate-direction air stream moved only the top leaves to and fro, revealing only some of the melons located under the vision field of the camera. This was partly due to the insufficient velocity of the air (up to 15m/s), blown across the melon bed and partly because the air stream emitted from the 200 mm wide (by 40 mm) outlets covered only part of the width of the approximate 800 by 800 mm field of view (FOV) of the camera. A test conducted using high velocity air streams from two Knapsack sprayers improved results. The air stream from these was swept alternately across the melon plants from the opposite sides of the bed. To obtain similar air velocities from the available blower installed on the mobile frame, the diameter of the air outlets was reduced to 50 mm, a quarter of their original dimensions. Air velocities of up to 60 m/s were obtained. The outlets were mounted on vertical hinges which enabled them to be pivoted at a 60° angle forward and backward relative to the length of the melon bed. The motion is induced by double acting air cylinders 19 mm diameter (3/4"), having a stroke of 76 mm (3"). The cylinders are controlled through 12 Volt solenoid valves, activated by microswitches, resulting in a continuous operation cycle, started and interrupted by means of a main switch.

The cylinders are operated alternately one stroke at a time, directing one air stream from one side across the bed. The end of the stroke of the first cylinder activates the one on the opposite side and so forth. This prevents the air stream from both sides to meet in mid-stroke, over the bed. The air-sweep across the foliage is started when the camera reaches its position over the next picking location. It is operated for one or two full 4 stroke cycles and then stopped. At the same time the air flow from the blower is opened or blocked by means of the butterfly valves installed in the blower outlets. The valves are actuated by an additional air cylinder and solenoid valve activated by the main switch. Stopping the air stream across the melon plants when not needed was considered necessary in order to minimize damage to the plants.

2.3.2.3. Light tent

During early field tests it was discovered that shadows of beams and other objects in the field of view of the camera, caused excessive noise for image processing. A light tent was constructed over the front portion of the mobile frame. For shading material, a double layer of silver colored polypropylene was used. The shading fabric was draped over an elevated beam mounted along the center of the superstructure so that it slopes down in both directions and covers the sides of the frame like a tent.

The light tent made of the double-layer, silver-colored, polypropylene netting did not diffuse sunlight well enough to eliminate all noise-causing shadows. In addition, sun rays penetrated the vision area through the uncovered manipulator section of the mobile frame. Thus, the complete mobile frame was covered with a light-diffusing material which eliminates distinct shadows, but leave a high enough light level for proper operation of the image sensors (Video cameras). The material chosen is a reinforced PVC heavy-gauge film of pink color, used to cover greenhouses. Pockets which retain structure rods and tie-down ropes were glued to the sheeting, which was draped over a new superstructure covering the whole mobile frame. This stands 2.35 m from ground level extending above the fully lifted Z axis column.

The pink, PVC material hangs down to 0.3 m above the ground level on the long sides of the frame to block direct rays during low sun elevations. At the front and rear ends, the drape extends down to the main frame. When the sun doesn't cast shadows, the PVC sheeting can be rolled up to increase ventilation and for convenient access.

2.3.2.4. Collecting troughs for harvested melons

Soft, tarpaulin troughs were installed along the bottom of each side of the frame, parallel to the X axis, and within the reach of the manipulator. These troughs simulate conveyers, onto which the grippers place harvested melons. Each trough frame was 190 cm long and 25 cm wide. A skin of man-made tarpaulin fabric is loosely fitted to the sides of the frame forming a trough about 10 cm deep. The position of the troughs is adjustable, relative to the ground level, and may be moved in and out relative to each other. The trough are hung on pivots which enable them to swing outward when pushed accidentally by the grippers.

2.3.2.5. Reflection-brightness-level references

Four reflection-brightness-level calibration plates, two white and two black pairs measuring 5 x 5 cm, were installed under the four extreme corners of the far-camera field of view. The vertical and horizontal positions of these plates are adjustable in order to change the distance from each other and their position on the X and Y axes relative to the camera axes.

2.3.2.6. Lighting fixtures

Mounting brackets for three, 500 W halogen light fixtures designed for outdoor use (*i.e.*, water tight seals) were installed in the vision section of the robotic frame. Two halogen lights were clamped to the arms holding the calibration plates and directed onto the vision field of the far camera. The third was positioned in the picking zone which the near camera covers by traveling along the X and Y axes.

2.3.2.7. Gripper design

The melon gripper consists of (Figure 32): the mounting frame, the gripping device and the cutting attachment, and is operated by pneumatic cylinders. The mounting frame is designed to be attached vertically to the lower end of the Z axis with the gripping device operating beneath the carriage. This frame accommodates the activating mechanisms, including two 3.81 cm (1.5") diameter pneumatic cylinders, one with a 2.54 cm (1") stroke to operate the gripper and one with a 5.08 cm (2") stroke to operate the cutter. Pneumatic activation was chosen over electric and hydraulic due to its light weight, cleanliness even when leaking, its rapid response, and ease of speed control. The gripping device was designed to restrain the melons rather than to grip them. It is made of a pair of hinged ring-jaws and two rounded, side-plates to form the outlines of a spherical retaining shape approximately 180 mm internal diameter (I.D.). The rings are 200 mm O.D. and are made of 8 mm diameter stainless steel rod. The hinges are spaced 68 mm. The rings close to a position parallel to each other, and pivot 18° to form an opening of 200 mm at the bottom of the gripper.

The vine-trimmer cutting attachment is made of two arms, one fitted with a shape blade and the other with a slotted anvil. The arms are attached at the center of the rounded, side-plates. The arms counter-rotate around the outside of the gripper rings, including the gap

between the jaw rings. Each rotates about 140° from the top, downwards to meet the opposite half of the cutter. Any vine trailing the melon, through the gap between the rings is cut between the blade and the slotted anvil. The cutting arms are cable-pulled and spring-returned to the top of the gripping jaws, and are out of the way when the melons are picked-up or released. This type of cutting device has the ability to cut vines and stems regardless of their location. Cutting between the blade and the anvil at the bottom of the gripper causes less pull on the stem by reducing the rotational movement of the melon and helps to ensure a clean cut. Cutting by impact was required because of the unfavorable leverage of the activating system where a short lever must operate a long arm.

In operation, the open gripper is aimed to reach centrally over the melon until the jaws touch the soil surface. This touch activates the bottom sensor. The Z column is lifted 2 to 3 cm, then the gripper is closed by activating a pneumatic cylinder. The melon is lifted from the ground, which stretches the stem, causing it to trail somewhere between the ring-jaws. Then the cutting attachment is activated to sever the vine, leaving a piece of it attached to the melon. Finally, the gripper is opened to release the melon onto the simulated conveyor or trough.

Figure 32. The Ring-Jaw Grippers

2.4 Field Tests

2.4.1 Field Procedures

Two different modes of operation were evaluated: step mode (off-line) and continuous mode (on-line). In the step mode, the vehicle advances along the row and stops at each batch of fruit. Then the robot arm approaches each fruit and picks the ripe ones. Once all ripe fruit in a location are picked, the vehicle advances to the next batch whose location is detected by the far-vision camera system. In the continuous mode, the robot arm moves towards the fruit and then follows it while the vehicle constantly moves along the row.

Three different experiments were conducted: imaging, fruit-picking and on-line mode tests. Field tests were conducted for two different melon cultivars, in two different seasons at two different locations: Ein Tamar (Jordan Valley, Israel; *cv Galia*) and Kibbutz Yifat (Israel Valley, Israel; *cv 14E*) in the spring (April) and summer (July) 1993 seasons.

The major imaging analysis consisted of black and white (B&W), gray level imaging (Dobrusin *et al.*, 1992). However, since only 80% of the fruit were detected in this mode, additional image processing techniques were evaluated: color, infrared and structured light. All imaging experiments were conducted in the continuous-mode of operation, *i.e.*, images were snapped while the vehicle was advancing along the melon bed. All analysis of enhanced image processing techniques were conducted independently, *i.e.*, with no connection to the robot operations.

All fruit-picking experiments were conducted in the step mode using black and white imaging. On-line mode tests were conducted in the laboratory to determine maximum performance which could be achieved. In these experiments, ideal visual conditions were utilized. This included elimination of foliage which may obscure fruit grown in the field, and employing uniformly-white colored targets to be grasped. During these experiments, objects (balls) were placed on the ground with different distances between them. One goal of the experiments was to define the minimum number of objects and the minimum distance between two adjacent targets which the robot can grasp without requiring the vehicle to stop. Another objective of these experiments was to determine the average time it takes to pick up a melon and transport it to a conveyer. These data would establish the relationships among vehicle forward speed, melon density, robot harvester rate, and melon harvest efficiency.

All on-line experiments were conducted for two different vehicle speeds 10 and 5 cm/sec; corresponding to the maximum and minimum values that can be achieved by the tractor, and for two different manipulator velocities (75 and 37.5 cm/sec). Thus, four types of experiments were conducted for each melon setup.

2.4.2 Performance results

2.4.2.1 Sensing Performance

Results indicate that gray-level and color-based algorithms have equivalent performances for the tested cultivars and approximately 80% of the fruit are detected (Table 12). The fruit dimension (the ratio between the fruit and its leaves diameters) is the critical parameter: detection results improve for bigger fruit. Non-detection cases were results of foliage occlusion (12%) and small melons, - less than 10 cm in diameter (6%). Several cases

of false detection were registered caused by big bright leaves and/or by bright patterns of the plastic mulch. The algorithm's execution time is acceptable since it is in the range of the estimated average robot harvesting cycle (Edan and Miles, 1991).

The laser system detected fewer melons. However, this was because the algorithm was not optimized for the real-time system and during the analysis, some of the image frames were missed because of the long computation time. Eventually, the laser system algorithm will be much faster than other imaging techniques (Benady and Miles, 1992). Therefore, these initial results from field studies must be considered preliminary.

A preliminary analysis was conducted on the infra-red imaging (Dobrusin, *et al.*, 1992) since no dedicated system was available for the research. This analysis indicated that temperature differentials of various objects in the field - melons, leaves, and the ground - were considerably dependent on the time of day and reached maximum values just after sunset. In contrast to images obtained in the visible spectrum - where leaves are the main source of errors in the location of the melon, it is not particularly difficult to distinguish a melon from the leaves in the infrared spectrum; this can be achieved by setting a threshold value. The source of errors in this case is represented by open ground patches which have temperatures commensurable with melons. Additional research is necessary to evaluate the full potential of infrared imaging for fruit location. Combining infrared imaging with visible-spectrum vision would likely improve melon detection results.

The color-based approach must be further investigated. Spectral differences between melons and leaves are variety dependent and sometimes leaves appear yellow, similar to a ripe fruit and thus lead to a false detection. High-quality color cameras must be used to improve the spectral content of images. In order to establish an accurate determination of ripeness which can be used for selectively harvesting, images must be color calibrated (Hetzroni, 1994). Color-based and (shape-based) gray-level approaches could be combined to improve detection results. This would require the development of sensor fusion techniques.

An adaptive segmentation technique must be developed. This can be achieved by a local independent segmentation of the image regions which are suspected to be ripe fruits; these regions must be determined before hand, based on their high average intensity. Artificial lighting and shading effects must be further studied. The impact of the air blast is significant and it's interaction with the vision requires more investigation. Hardware synchronization of the vision and the air blast could improve melon detection.

2.4.2.2 On-line mode testing

The minimum distance between sequential melons that enable the robot to advance along the row without stopping is indicated in Table 13. The minimum distance declines as forward speed decreases. However, the average pick time does not decrease proportionally due to the delays caused by the acceleration and deceleration at the beginning and end of the movement.

When the melons were placed with a distance smaller than the minimum, the robot controller generated a stop signal and the forward speed of the vehicle ceased. Forward movement of the vehicle resumed automatically after all ripe melons in the manipulator workspace were picked. In initial versions of the control program, the stopping algorithm was not implemented, and the robot would miss melons when their spacing was too close.

An experiment with the same melon setup (positions) was conducted also for the off line mode. Since vehicle velocity does not affect the overall performance, all experiments were conducted for a speed of 10cm/sec. Results are summarized in Table 14. In general, the average cycle time decreases as the robot's actuator speed increases. The average picking time is very large (40 s) when the distance between sequential melons is 50 cm. In this case, when the robot moves to get a melon, the next melon exits the far camera field of view and can not be located. Therefore, the vehicle must move back to place the melon into the far sensor's workspace. With this melon setup, the on-line operation mode is 3-4 times faster than the off-line mode.

Melons usually grow in hills 2-4 fruit per cluster (Edan and Miles, 1993). When melons are located, the vehicle must stop to permit the robot time to harvest all the ripe fruit. Table 15 presents the results obtained for different melon clusterings. The best result (8 sec per melon) was achieved at a vehicle velocity of 5 cm/sec, which is half the maximum speed. Excessive speeds of the vehicle created a need to stop in order to harvest all the fruit in a cluster, which reduced field capacity (fruit per hour, or area per hour). This proves that the optimum velocity parameters are not always the highest ones. The average pick time depends on both the robot and manipulator speeds. When the robot must stop (in order not to miss melons) the average picking time is hardly increased since most of the time is "wasted" during the stop.

2.4.2.3 Fruit picking

In addition to the many field studies conducted for general operation validation (part of which were documented in a videotape), specific experiments were conducted to evaluate system performance. In each field approximately 100 fruit were analyzed which included the following evaluations: number of fruit detected by the far camera; number of fruit detected by the eye-in-hand camera; number of loops; number of fruit reached by the manipulator; number of fruit successfully picked with one cut; number of fruit requiring a 2nd cut (which was added for the experiments conducted in the summer since the first field tests indicated that the thick stems were not all severed with just one cut); number of fruit that were snapped off (causing damage to the melon) and the number of false detection of fruit. Results for all four fields is presented in Appendix A, Tables A1 - A3. Table 16 summarizes the results of each experiment.

Overall, 93.57% of the fruit were successfully detected by the near camera. 90.54% of the fruit were actually reached by the robot arm; 85.67% were successfully picked; 5.85% required a second cut attempt; and 7.02% were disconnected from the stem in such a way that would cause damage to the fruit. Most of the 19.88% false picks (*i.e.*, the robot performed the whole pick loop with no fruit present) were caused by the plastic mulch which covered the ground. This decreased significantly during the summer when the experiments were conducted on dark brown ground with no plastic mulch.

Table 12. Summary of vision and laser system detection results

Detection Method	Detected Fruit (%)	Execution Time (Sec/image)
Gray-level	82 - 87	1.1
Color	82 - 88	1.3
Laser	74	5.5

Table 13. Experimental data for straight line melon setup (On-line)

Vehicle Speed (cm/sec)	Robot Speed (cm/sec)	Minimum Distance (cm)	Average Time (sec/melon)
10	75	50	7
5	75	25	7
10	37.5	70	9
5	37.5	45	9

Table 14. Experimental data for straight line melon setup (Off-line)

Vehicle Speed (cm/sec)	Robot Speed (cm/sec)	Minimum Distance (cm)	Average Time (sec/melon)
10	75	50	40
10	37.5	50	45
10	75	100	23
10	37.5	100	25

Table 15. Experimental data for cluster melon setup (On-line)

Vehicle Speed (cm/sec)	Robot Speed (cm/sec)	Distance (cm)	Melons per cluster	Average Time (sec)	Stop
10	75	100	2	15	Yes
10	37.5	100	2	17	Yes
5	75	100	2	8	No
5	37.5	100	2	19	Yes
10	75	100	3	16	Yes
10	37.5	100	3	18	Yes
5	75	100	3	19	Yes
5	37.5	100	3	22	Yes

Table 16. Summary of fruit-picking field experiments

Experiment	Far	Near	Reached	Picked	False pick	Actual
1	94	69	71	68	31	63
2	112	100	101	97	19	104
3	88	74	74	67	4	83
4	97	77	64	61	14	92
Total	320	259	310	293	68	342

2.5 References

- Abeles, F. B. 1973. Ethylene in plant biology. Academic Press, N.Y.
- Altman, S. A. and K. Corey. 1986. Resistance to gas diffusion through the rind and flesh of two muskmelon subspecies. HortScience, 21:858 (Abstr No. 1443).
- Aubert, S. and R. Dumas de Vaulx. 1977. Undestructive firmness test for the maturity control of melons. Ann. Technol. Agric., 27(3):243-254.
- Baffes, P. T. 1991. Nets User's Manual Version 3.0. AIS at NASA/JSC, Athens, Georgia.
- Ballard, D. H. and C. M. Brown. 1982. *Computer vision*. Prentice Hall.
- Benady, M. and G. E. Miles. 1992. Locating melons for robotic harvesting using structured light. ASAE Paper No.92-7021, ASAE, St. Joseph, MI 49085.
- Benady, M., Simon, J.E., Miles, G.E. 1992. Determining melon ripeness by analysis of headspace gases emissions. ASAE Paper No.92-6055, ASAE, St. Joseph, MI 49085..
- Benady, M., Y. Edan, A. Hetzroni, and G. E. Miles. 1991. Design of a field crops robotic machine. American Society of Agricultural Engineers, St. Joseph, MI. Paper No. 91-7028.
- Benady, Meny. 1993. Sensory perception for an agricultural robot. PhD Dissertation. Department of Agricultural Engineering. Purdue University. West Lafayette, IN 47907-1146. 278 p.
- Bliss, M. L. and H. K. Pratt. 1979. Effect of ethylene, maturity, and attachment to the parent plant on production of volatile compounds by muskmelons. J. Amer. Soc. Hort. Sci., 104(2):273-277.
- Bolte, J. P. 1989. Applications of neural networks in agriculture. Paper No. 89-7591. The ASAE, St. Joseph, MI 49085-9659.
- Brandon, J. R., M. S. Howarth, and S. W. Searcy. 1990. A neural network for carrot tip classification. Paper No. 90-7549. The ASAE, St. Joseph, MI 49085-9659.
- Brons, A., G. Rabatel, F. Sevilla, and C. Touzet. 1992. Neural network techniques to simulate human judgement on potplant quality: an artificial expert for beauty classification. Proceedings of 1991 Symposium of Automated Agriculture for the 21st century. American Society of Agricultural Engineers. pp 126-133.
- Cantwell, M. 1990. Honeydew, orange-flesh and cantaloupe melon quality. Report to the California Melon Advisory Board.
- Cardenas-Weber, Martha, Richard L. Stroshine, Kamyar Haghighi and Yael Edan. 1989. Finite element analysis of melon handled by different robotic gripper designs. Paper No. 89-1602. The ASAE, St. Joseph, MI 49085-9659. 20p.
- Cardenas-Weber, Martha, Richard Stroshine, Kamyar Haghighi, and Yael Edan. 1991. Melon materials properties and finite element analysis of melon compression with application to robot gripping. Transactions of ASAE 34(3):920-929.
- Cardenas-Weber, M., Hetzroni, A., Miles, G. E. 1991. Machine vision to locate melons and guide robotic harvesting. ASAE Paper No.91-7006, ASAE, St. Joseph, MI 49085.
- Cardenas-Weber, Martha A. 1991. Design and testing of a machine vision system for a robotic harvester of melons. PhD Dissertation. Department of Agricultural Engineering. Purdue University, West Lafayette, IN 47907-1146. 424 p.
- Charles, D. J. and J. E. Simon. 1990. Comparison of extraction methods for the rapid determination of essential oil content and composition of basil (*Ocimum* spp.). J. Amer. soc. Hort. Sci., 115:458-462.

- Cho, S. I. and G. W. Krutz. 1989. Fruit ripeness detection by using ^1H NMR. American Society of Agricultural Engineers, St. Joseph, MI. Paper No. 89-6620.
- Cho, S. I., W. K. Wai, G. W. Krutz and R. L. Stroshine. 1991. Prototype of non-destructive fruit sweetness sensor. Paper No. 91-3524. The ASAE, St. Joseph, MI. 49085-9659.
- CLIPS. 1989. Reference Manual. Version 4.3. Artificial Intelligence Section. Johnson Space Center, TX.
- Dobrusin, Yuri, Yael Edan, Josef Grinshpun, Uri M. Peiper, and Amots Hetzroni. 1992. Real-time image processing for robotic melon harvesting. Paper No. 92-3515. The ASAE, St. Joseph, MI. 49085-9659.
- Duda, R. O. and P. E. Hart. 1972. Use of the Hough transform to detect lines and curves in pictures. *Commun. ACM*. 15(1).
- Dull, G. G. 1986. Nondestructive evaluation of quality of stored fruits and vegetables. *Food Tech.*, pp. 106-110.
- Dull, G. G., R. G. Leffler, G. S. Birth, and D. A. Smittle. 1990. Instrument for non-destructive measurement of soluble solids in honeydew melons. American Society of Agricultural Engineers, St. Joseph, MI. Paper No. 90-3553.
- Edan, Y. and G. E. Miles. 1989. Animated simulation of robotic melon harvesting. Paper No. 89-76612. The ASAE, St. Joseph, MI. 49085-9659. 9 p.
- Edan, Yael, Kamyar Haghighi, Richard L. Stroshine, and Martha Cardenas-Weber. 1989. Finite element analysis and optimization of a robotic gripper design. Paper No. 89-7537. The ASAE, St. Joseph, MI. 49085-9659. 14p.
- Edan, Y. 1990. Control and Design of an Intelligent Agricultural Robot. Ph.D Thesis, Purdue University, West Lafayette, IN 47907. 207 p.
- Edan, Yael, Bernard A. Engel, Gaines E. Miles, and Uri M. Peiper. 1991. Intelligent control of a robotic melon harvester. Proc. Second France-Israel Symposium on Robotics: Advances in non-manufacturing robotics. April 8-10, Saclay, France. Institut National des Sciences et Techniques Nucleares. pp 6.33-6.47.
- Edan, Y. and G. E. Miles. 1991. Design of an agricultural robot for harvesting melons. American Society of Agricultural Engineers, St. Joseph, MI. Paper No. 91-7029.
- Edan, Yael and Shimon Y. Nof. 1992. Robot Motion Economy Analysis and Program. Research Memorandum No. 92-5. School of Industrial Engineering. Purdue University, West Lafayette, IN. 47906. 30p.
- Edan, Y. and G. E. Miles. 1993. Design of an agricultural robot for harvesting melons. *Transactions of the ASAE* 36(2):593-603.
- Edan, Yael, Bernard A. Engel and G. E. Miles. 1993. Intelligent control system simulation of an agricultural robot. *Journal of Intelligent and Robotic Systems*. 8:267-284.
- Engel, B. A. and X. Zhuang. 1991. Interpreting satellite thematic mapper data using neural networks. Proceedings of 1991 Symposium of Automated Agriculture for the 21st century. American Society of Agricultural Engineers. pp 144-154.
- Engelmore, R. and T. Morgan. 1988. Blackboard systems. Addison Wesley. Wokingham, England.
- Farabee, M. L. and M. L. Stone. 1991. Determination of watermelon maturity with sonic impulse testing. American Society of Agricultural Engineers, St. Joseph, MI. Paper No. 91-3013.
- Figaro. 1990a. FIGARO Gas Sensor TGS 822. Figaro Engineering Corporation. Osaka, Japan.

- Figaro. 1990b. TGS - Semiconductor Gas Sensor. Figaro Engineering Corporation, Osaka, Japan. 18 p.
- Flash, T. 1991. Biological models in robotics. Second France-Israel Symposium on Robotics: 5.15-5.28.
- Gonzalez, R.C. and Wintz, P. 1987. Digital image processing. Second edition. Addison Wesley Publ. Co., MA.
- Haralick, R.M., Sternberg, S.R. and Zhuang, X. 1987. Image analysis using mathematical morphology. IEEE Trans. in Pattern Analysis and Machine Intelligence 9:523-550.
- Haralick, R.M. and Shapiro, L.G. 1985. Survey: Image Segmentation Techniques. Computer Vision Graphics and Image Processing 29:100-132.
- Hetzroni, Amots. 1994. Machine vision monitoring of plant nutrition. PhD Dissertation. Department of Agricultural Engineering. Purdue University. West Lafayette, IN 47907-1146. 263 p.
- Horvat, R. J. and S. D. Senter. 1987. Identification of additional volatile compounds from cantaloupe. J. Food Science, 52(4):1097-1098.
- Hough, P. V. C. 1962. Method and means for recognizing complex patterns. U.S. Patent 3,069,654. Dec. 18.
- Kader, A. A., R. F. Kasmire, F. G. Mitchell, M. S. Reid, N. F. Sommer, and J. F. Thompson. 1985. Postharvest technology of horticultural crops. Technical report, Univ. of Calif.-Davis, Coop. Ext. Serv. Spec. Pub. 3311.
- Kasmire, R. F. 1977. Electronic color sorting of cantaloupes for ripeness. Calif. Agric., p. 22.
- Kasmire, R. F. 1981. Muskmelon production in California. Univ. of Calif. - Davis. C.E.S. Leaflet 2671, 23 p.
- Kemp, T. R., D. E. Knavel, and L. P. Stoltz. 1973. Volatile cucumis melo components: Identification of additional compounds and effects of storage conditions. Phytochemistry, 12(12):2921-2924.
- Kemp, T. R., L. P. Stoltz, and D. E. Knavel. 1972. Volatile components of muskmelon fruit. J. Agr. and Food Chemistry, 20(2):196-198.
- Kendall, S. A. and T. J. Ng. 1988. Genetic variation of ethylene production in harvested muskmelon fruits. HortScience, 23(4):759-761.
- Kitamura, T., T. Umemoto, T. Iwata, and T. Akazawa. 1976. Studies on the storage of melons, III: Varietal differences in the fluctuation of the amount of volatile constituents produced during ripening. J. Jap. Soc. for Hort. Sci., 44(4):417-421.
- Klein, C. A. and Maciejewski, A. A. 1988. Simulators, Graphic. IN: R.C. Dorf and S. Y. Nof (Eds). International Encyclopedia of Robotics, Automation and Applications. John Wiley and Sons, Inc., New York. pp 1599-1608.
- Leopold, A. C. and P. E. Kriedemann. 1975. Plant Growth and Development. McGraw Hill Book Company.
- Miles, Gaines E. and You-Jen Tsai. 1987. Combine systems engineering by simulation. Transactions of the ASAE 30(5):1277-1281.
- Miles, Gaines E. 1990. Robotic Systems for Selective Harvesting: Basic Concepts and Prototype Tests. Final Report BARD Project No. US-1254-87. Department of Agricultural Engineering. Purdue University. West Lafayette, Indiana. 47906-1146. 176 p.
- Miller, W. M. 1992. Classification analysis of Florida grapefruit based on shape parameters. Proceedings of Food Processing Automation II, Lexington, KY. pp. 339 - 347.

- Miyazaki, T. and M. Ookubo. 1989. Effects of maturity and postharvest techniques on keeping quality of melons. *J. Jap. Soc. Hort. Sci.*, 58:361-368.
- Mizrach, A., N. Galili, D. C. Teitel, and G. Rosenhouse. 1989. Acoustical, mechanical and quality parameters of melon tissue. American Society of Agricultural Engineers, St. Joseph, MI. Paper No. 89-3051.
- Moravec, H.P. 1990. The Stanford Cart and the CMU Rover. In: *Autonomous robot vehicles*, I.J. Cox and G.T. Wilfong. Springer-Verlag, New York: 407-419.
- Morrison Baird, L. A. and B. D. Webster. 1979. The anatomy and histochemistry of fruit abscission. *Horticultural Review*, 1:172-203.
- Nii, H. P.. 1987. Blackboard systems: The blackboard model of problem solving and the evolution of blackboard architectures. *The AI Magazine* 7(3):38-53.
- Nof, S. Y. 1985. Robot ergonomics: optimizing robot work. IN: S. Y. Nof (Ed). *Handbook of Industrial Robotics*. John Wiley and Sons, Inc., New York.
- Nof, S. Y. and P. Robinson. 1986. Analysis of two robot motion economy principles. *Israel Journal of Technology*. 23:125-128.
- Nof, S. Y. 1991a. Advances in robot motion economy principles. *Proceedings of ISIR 22*, Detroit, MI. 22_13 - 22_21.
- Nof, S. Y. 1991b. *Industrial Robotics*. IN: G. Salvendy (Ed). *Handbook of I.E.*, John Wiley and Sons, Inc., New York.
- Optotrak, 1990. 3D motion Measurement System, Technical Production Description. Northern Digital Inc., Waterloo, Ontario, Canada.
- Pasternak, H., A. Lidror, and H. Engel. 1989. The effect of the incidence of defect on orange inspection time. *Canadian Agricultural Engineering*, 31(2):131-134.
- Pratt, H. K. 1971. Melons. In: Hulme, A. C., (editor), *The biochemistry of fruits and their products*, volume 2, pp. 207-232. Academic Press, London.
- Reid, M. S. 1985. Ethylene and abscission. *HortScience*, 20:45-50.
- Reid, M. S., T. H. Lee, H. K. Pratt, and C. O. Chichester. 1970. Chlorophyll and carotenoid changes in developing muskmelons. *J. Amer. Soc. Hort. Sci.*, 95:814-815.
- Rigney, M. P. and G. A. Kranzler. 1989. Seedling classification performance of a neural network. Paper No. 89-7523. The ASAE. St. Joseph, MI. 49085-9659.
- Rosenfeld, A. and Kak, A.C. 1982. *Digital picture processing*. Academic Press, New York.
- Rumelhart, D. E., G. E. Hinton and R. J. Williams. 1986. *Parallel distributed processing*. MIT Press. Cambridge, Massachusetts. 547 p.
- Simon, J. E., D. R. Decoteau, and L. E. Craker. 1983. Rapid detection of phytotoxic compounds using stress-induced ethylene and ethane. *Environ. Tech. Letters*, 4:157-162.
- Thai, C. N., J. N. Pease, and E. W. Tollner. 1991. Determination of green tomato maturity from X-ray computed tomography images. IN: *Proceedings of the 1991 Symposium of Automated Agriculture for the 21st Century*. American Society of Agricultural Engineers. pp 134-143.
- Tseng, A. A. 1988. Assessment of robotic simulation software. *Proceedings of the Computers in Engineering Conference 2*. ASME. pp 217-226.
- Upchurch, B. L., G. E. Miles, R. L. Stroshine, E. S. Furgason, and F. H. Emerson. 1987. Ultrasonic Measurements for Detecting Apple Bruises. *Trans. ASAE* 30(3):803-809.
- Whittaker, A. D., B. S. Park, and J. D. McCauley. 1991. Ultrasonic signal classification for beef quality grading through neural networks. *Proceedings of the 1991 Symposium of*

- Automated Agriculture for the 21st century. American Society of Agricultural Engineers. pp 116-125.
- Wolf, Itzhak, Jonathan Bar-Or, Yael Edan, and Uri M. Peiper. 1990. Developing grippers for a melon harvesting robot. Paper No. 90-7504. The ASAE, St. Joseph, MI. 49085-9659. 9p.
- Yabumoto, K., W. G. Jennings, and M. Yamaguchi. 1977. Volatile constituents of cantaloupe, cucumis melo, and their biogenesis. J. of Food Science, 42:32-37.
- Yabumoto, K., M. Yamaguchi, and W. G. Jennings. 1978. Production of volatile compounds by muskmelon, cucumis melo. Food Chemistry, 3(1):7-16.
- Zhuang, Xin, Bernard A. Engel, and Meny Benady. 1992. Locating melons using artificial neural networks. Paper No. 92-7014. The ASAE, St. Joseph, MI. 49085-9659. 11 p.
- Zhuang, X., B. A. Engel., J. Jia, and A. P. Schinckel. 1992. Potential applications of artificial neural networks in biological systems. EurAgEng Paper No. 9206-04, IN: Proceedings of AgEng '92 International Conference on Agricultural Engineering. Uppsala, Sweden. pp 345-346.
- Zhuang, X. and B. A. Engel. 1990a. Neural networks for applications in agriculture. Paper No. 90-7024. The ASAE, St. Joseph, MI. 49085-9659.
- Zhuang, X. and B. A. Engel. 1990b. Classification of multispectral remote sensing data using neural networks vs statistical methods. Paper No. 90-7552. The ASAE, St. Joseph, MI. 49085-9659.

3. EVALUATION OF RESEARCH ACCOMPLISHMENTS AND CONCLUSIONS

This project has successfully completed both its general and specific objectives. The general objective of this work was to integrate the melon gripper, machine vision sensing of fruit location, and robotic motion simulations into a field prototype machine. The specific objectives were:

1. To improve and extend the fundamental technologies of gripping, sensing, and manipulator control;
2. To integrate the individual, intelligent components (the robot controller and the image processor) under a central control processor; and
3. To design, construct and field-test a prototype robotic melon harvester.

The advancements in technologies to sense and selectively harvest easily-bruised fruit are highly significant and have multiple applications in other arenas. To some, the final prototype may appear infantile, but what critics don't realize is that to run, one must first learn to crawl, stand, and walk. The VIP-ROMPER isn't running 4-minute miles, but it is definitely locomoting and accomplishing its intended purpose: selectively harvesting melons. Advances in computing since the purchase of the HP-486, 25 Mhz machines 3 years ago, would now provide 3-5 times the computational capacity for less money. Who would have believed that the cutterbar on the reaper invented by Cyrus McCormick more than 100 years ago would still be in use on today's modern combines? This project has shown what it takes to selectively harvest melons and in doing so, has laid the foundation for many future applications of robotics in agriculture.

In addition to these general accomplishments, specific advances were made in sensors, intelligent controls, and mechanical design of agricultural robots. Two necessary tasks for selective harvesting, sensing of fruit location and sensing of fruit ripeness, were accomplished with a combined machine vision and laser imaging system, and an electronic "sniffer". With machine vision and image processing alone, from 82 to 88% of the melons were detected; however, false detections (due to leaves or bright spots on the soil surface) occurred as often as 36% of the time. Obviously, this would reduce the throughput by 36%. The addition of a laser plane projector reduced the number of false detections to approximately 2%. The additional laser sensor provides 3-D depth and ranging information, but requires extra hardware to perform the processing in real-time. Ripeness was determined by color of the fruit, and by electronically measuring the aromatic volatiles emitted by muskmelons as they mature. The SnO₂ sensor output correlated well with other measures of melon ripeness, but the instrument used for laboratory analyses was too slow to implement on the field prototype. Additional work will be necessary to shorten the response time of the SnO₂ sensor and permit it to be used to direct the robot to harvest only the ripe fruit. This sensor has great potential for on and off line grading in packing houses, especially when integrated with other measures of ripeness and quality.

Unlike industrial robots whose paths are predictable and pre-programmable, agricultural robots must be able to perceive and respond to a dynamic environment. This requires close communication among the processors which analyze data from sensors, and computers which control motions of the manipulators. The development of an intelligent control strategy and its integration with multiple, complex sensors is a significant

accomplishment made possible via the integration of a Personal Computer (HP-486) as the executive, a real-time image co-processor, and a real-time robot control card. This distributed processing concept provided the cost/performance ratio required for successful development of the prototype robotic harvester.

The design criteria for reach, mass, speed, acceleration, and torque exceeded the performance specifications for any commercial robot; thus this project designed a unique machine, using off the shelf components when possible. The mechanical components of each robotic axis represent a truly imaginative, efficient, and effective design. It is flexible and can be adapted to many other applications. Electrical power was chosen for this project, but future robots based on this design will be able to use hydraulics provided by the tractor which tows the frame. Other advances in technologies include a flexible wrist, and clam-shell or ring-type grippers which grasp the melons, without damage. A knife which trims the vine from the fruit is also a unique innovation.

When evaluating the successes of this project, one cannot ignore the contributions of the simulation and systems analysis efforts. Not often is the design of a machine thoroughly analyzed before it is constructed. But, because robots are so flexible, there are many alternative designs. By modeling proposed configurations and simulating their performances for various field, crop, and machine parameters, the best designs became apparent, and were chosen to be the basis of the field prototype. Without the simulations, field studies could have become a never-ending process of tests, analyses, and modifications. With simulations, changes in machine performance can clearly be attributed to improvements in design, or to changes in crop or field conditions. By utilizing the knowledge gained during the simulations, the field studies avoided pitfalls in the design.

Advances in sensors, control systems, and hardware enabled the prototype selective harvester to perform as designed during field tests. Data gathered during these evaluation tests prove that the fundamental concepts of robotic harvesting are valid, and that robots and derivatives (robotics) have many potential applications in agriculture. The prototype agriculture robot can also be used to harvest and possibly transplant other "head" crops such as pumpkins, lettuce, broccoli, cauliflower, cabbage, etc. The machine vision and image processing sensors have other applications such as site-specific application of pesticides directed onto plants. Work with the SnO_2 sensor has opened the door for electronic monitoring of dangerous levels of gases and chemicals in controlled environments. Thus, the significance of this work isn't just the data on the field performance of a selective harvester for melons. It's real contribution to the body of scientific knowledge is much broader and deeper.

4. COOPERATION

The amount and level of cooperation among the researchers at the participating institutions has been unparalleled. The complicated nature of this research required close, coordinated efforts which were made possible by BARD funds for international travel and for regular electronic communications via internet. The work plan called for the basic design to be completed at Purdue, and for the construction and field-testing of the prototype to be done in Israel. This was successfully accomplished via the diligent efforts of dedicated engineers, scientists, and technicians.

First of all, the prototype frame and mechanical components of the robotic axes were designed, constructed and tested by I. Wolf at the Volcani Center. During a visit to Purdue, he was able to evaluate a commercial robot proposed to the project, and determine that it was inadequate. Upon his return to Israel, I. Wolf completed the mechanical component design and forwarded it to Purdue.

Secondly, the robotic controls and sensors were purchased, interfaced to the HP-486 computer, programmed, and integrated by Meny Benady, Amots Hetzroni, Yael Edan and others at Purdue under the direction of Dr. Gaines E. Miles. After confirming the success of the integration in lab tests, the hardware and software were shipped to the Volcani Institute for installation on the prototype frame. This was made necessary due to insufficient funds to construct two machines, one in each country. Following a brief Post-Doc at Purdue, Yael Edan returned to the Volcani Institute where she lead the effort to complete the development of the prototype. Due to the knowledge gained via her prior experiences at Purdue and her extremely diligent efforts at Volcani, the prototype robotic melon harvester was successful. In the case of this most complicated technology, success required one person (Y. Edan) to participate in activities in both countries.

Brief visits to Israel by Meny Benady and Amots Hetzroni enabled knowledge gained at Purdue to be shared with colleagues in Israel. In the Spring of 1993, Dr. Miles visited Israel to deliver the hardware and software for the laser sensor. During this time, he assisted field testing of the prototype and suggested improvements in sensor calibration procedures.

The foundation for the success of this project was laid by theoretical efforts at both Purdue and the Weizmann Institute of Science. Once again, Yael Edan was the catalyst, floating between the investigators at Purdue, Weizmann, and Volcani, making sure work progressed towards a successful conclusion.

Internet communications have proven to be a most valuable means of cooperation. Not only have investigators sent email to each other on a regular basis (often several times a day), but accounts were established at Purdue which enabled researchers to login and transfer data files, programs, and text files.

It is the opinion of the principal investigator that the cooperation among investigators during this project is precisely the intent of the Binational Agricultural Research and Development Fund.

5. PUBLICATIONS

5.1 Articles in Refereed Journals

- Cardenas-Weber, M., R. Stroshine, K. Haghighi and Y. Edan. 1991. Melon material properties and finite element analysis of compression with application to robot gripping. Transactions of the ASAE 34(3): 920-929. (1 of 10 to receive "Best Paper of the Year" Award from ASAE).
- Edan, Y., K. Haghighi, R. Stroshine and M. Cardenas-Weber. 1992. Robot gripper analysis: Finite element modeling and optimization. Applied Engineering in Agriculture 8(4): 563-570.
- Edan, Y. and G.E. Miles. 1993. Design of an agricultural robot for harvesting melons. Transactions of the ASAE 36(2): 593-603.
- Edan, Y., B.A. Engel and G.E. Miles. 1993. Intelligent control system simulation of an agricultural robot. Journal of Intelligent and Robotic Systems. 8:267-284.
- Edan, Y. and S.Y. Nof. 1994. Motion economy analysis for robotic kitting tasks. International Journal of Production Research. (In Press)
- Edan, Y. and G. E. Miles. 1994. Systems engineering of agricultural robot design. IEEE Transactions on Systems, Man and Cybernetics. 24(8): (In Press).
- Edan, Y. 1994. Design of an Autonomous Agricultural Robot. The International Journal of Applied Intelligence. Special Issue on Autonomous Systems. (In Press).

5.2 Book Chapters

- Dobrusin, Y., Y. Edan, J. Grinspun, U.M. Peiper, I Wolf, A. Hetzroni. 1993. Computer image analysis to locate targets for an agricultural robot. Computer Analysis of Images and Patterns, D. Chetverikov and W.G. Kropatsch (Eds.), Springer Verlag, 5th Intl. Conference, Lecture Notes in Computer Science 719.
- Edan, Y., G.E. Miles and S.Y. Nof. 1993. Graphic-Based simulation and analysis of robotic systems. The International Handbook on Robot Simulation Systems.
- Edan, Y., B. Engel, G.E. Miles and U.M. Peiper. 1990. Intelligent control of a robotic melon harvester. Revue Automatique Productique Appliquees 3(3), 1990: 37-53. Editor: Hermes, France.

5.3. Dissertations

- Benady, Meny. 1993. Sensory perception for an agricultural robot. Ph.D Dissertation. Department of Agricultural Engineering, Purdue University. West Lafayette Indiana 47907-1146. 278 p.
- Cardenas-Weber, Martha. 1991. Design and testing a machine vision system for a robotic harvester of melons. Ph.D Dissertation. Department of Agricultural Engineering, Purdue University. West Lafayette Indiana 47907-1146. 424 p.
- Edan, Yael. 1990. Control and design of an intelligent agricultural robot. Ph.D Dissertation. Department of Agricultural Engineering, Purdue University. West Lafayette Indiana 47907-1146. 207 p.
- Rogozin, Vadim. 1993. Robotic Control of Dynamic Tasks. M.Sc. Dissertation. Department of Applied Mathematics and Computer Science, Weizmann Institute of Science. Rehovot 76100, Israel.

5.4. Papers and Proceedings Presented at National and International Conferences

- Benady, M., Y. Edan, A. Hetzroni and G.E. Miles. 1991. Design of a field crops robotic machine. Paper No. 91-7028, ASAE, St. Joseph, MI 49085. 7 p.
- Benady, M. J.E. Simon, D.J. Charles and G.E. Miles. 1992. Determining melon ripeness by analyzing headspace gas emissions. Paper No. 92-6055, ASAE, St. Joseph, MI 49085. 13 p.
- Benady, M. and G.E. Miles. 1992. Locating melons for robotic harvesting using structured light. ASAE Paper No. 92-7021, ASAE, St. Joseph, MI 49085. 14p.
- Cardenas-Weber, M., R. Stroshine, K. Haghighi and Y. Edan. 1989. Finite element analysis of melons handled by different robot gripper designs. ASAE Paper No. 89-1602, ASAE, St. Joseph, MI 49085.
- Cardenas-Weber, M., A. Hetzroni and G.E. Miles. 1991. Machine vision to locate melons and guide robotic harvesting. ASAE Paper No. 91-7006, ASAE, St. Joseph, MI 49085. 21 p.
- Dobrusin, Y., Y. Edan, J. Grinspun, A. Hetzroni and U.M. Peiper. 1992. Real-time image processing for robotic melon harvesting. ASAE Paper No. 92-3515 ASAE, St. Joseph, MI 49085. 16 p.
- Dobrusin, Y. and Y. Edan. 1992. Computer vision for robotic melon harvesting. 9th Israeli Symposium on Artificial Intelligence and Computer Vision.
- Edan, Y. and G.E. Miles. 1989. Animated, visual simulation of robotic melon harvesting. ASAE Paper No. 89-7612, ASAE, St. Joseph, MI 49085. 9 p.
- Edan, Y., K. Haghighi, R. Stroshine and M. Cardenas-Weber. 1989. Finite element analysis and optimization of a robot gripper design. ASAE Paper No. 89-7537, ASAE, St. Joseph, MI 49085. 14 p.
- Edan, Y., B. Engel, G.E. Miles, U.M. Peiper. 1991. Intelligent control of a robotic melon harvester. Second France-Israel Symposium on Robotics: 6.33-6.47.
- Edan, Y. and G.E. Miles. 1991. Design of an agricultural robot for harvesting melons. ASAE Paper No. 91-7029, ASAE, St. Joseph, MI 49085. 23 p.
- Edan, Y., M. Benady and G.E. Miles. 1992. Economic analysis of robotic melon harvesting. ASAE Paper No. 92-1512, ASAE, St. Joseph, MI 49085. 22 p.
- Edan, Y. Systems Engineering of agricultural robot design. The 1993 Operations Research Society of Israel National Meeting, Nahariya.
- Edan, Y., I. Wolf, V. Rogozin, J. Grinspun, Y. Dobrusin. 1994. Robotic Melon Harvesting: Prototype and Field Tests. Paper No. 94-3073. The ASAE, St. Joseph, MI. 49085.
- Rogozin, V., Y. Edan and T. Flash. 1993. Robotic control of dynamic tasks. The 1993 International Conference on Control Theory and Its Applications, Maale HaChamisha.
- Wolf, I., J. Bar-Or, Y. Edan and U.M. Peiper. 1990. Developing grippers for a melon harvesting robot. Paper No. 90-7504, The ASAE, St. Joseph, MI 49085. 9 p.
- Zhuang, Xin, B. Engel and M. Benady. 1992. Locating melons using artificial neural networks. ASAE Paper No. 92-7014, ASAE, St. Joseph, MI 49085. 11 p.

5.5 Manuscripts Submitted to Refereed Journals

- Benady, M. and G. E. Miles. Fruit Location in 3-D Using Structured Light, Journal Paper No. 13841, Purdue University, Ag. Experiment Station, West Lafayette, IN 47907. Submitted to Transactions of the ASAE. 1993.

Benady, M., J. E. Simon, D. J. Charles and G. E. Miles. Fruit Ripeness Determination by Electronic Sensing of Aromatic Volatiles, Journal Paper No. 13839, Purdue University, Ag. Experiment Station, West Lafayette, IN 47907. Submitted to Transactions of the ASAE. 1993.

Edan, Y. and S.Y. Nof. 1993. Graphic-based analysis of robot motion economy principles. Submitted to Robotics and Computer Integrated Manufacturing.

5.6 Departmental Reports

Edan, Yael and Shimon Y. Nof. 1992. Robot Motion Economy Analysis and Program. School of Industrial Engineering, Purdue University, West Lafayette, IN 47907. Research Memorandum No. 92-5. 30 p.

Edan, Y. and G. E. Miles. 1992. Design and Control of an Agricultural Robot. Agricultural Research Organization Publication 251. Summary of Research Activities 1988-1992.

Edan, Y. and Shimon Y. Nof. 1992. Motion Economy Analysis for Robotic Kitting Task. School of Industrial Engineering, Purdue University, West Lafayette, IN. 47907. Research Memorandum No. 92-21.

5.7 Citations and Articles in the Popular Press

- *American Vegetable Grower*. June. 1994. p 51. Harvesting Robot Senses Ripe Melons.
- *Boardroom Reports*. January 1, 1994. p 10. High-Tech Breakthroughs: Agricultural Robot.
- *Discover*, December 1993. p 26. Careful With That Fruit, Brute.
- *Farm Profit*. Fall, 1993. p 14. Ag Robots Will Soon Harvest Your Fruits, Vegetables.
- *World Book Annual Science Supplement*, 1994. pp 203-204. A Review of Science and Technology During the 1993 School Year: Agriculture.
- *Purdue Agriculture*. 1993-94. p 27. Choose 'n Pick
- *Christian Science Monitor*. December 1, 1993. p 8. Farming Takes on a New Era of Technology.
- *BARD IS/US Newsletter*. June 1993. p2. Robotic Melon Harvester.
- *Connections*, Spring 1993. p 8. Robots That Will Take to the Fields.
- *The Purdue Exponent*. January 21, 1993. PU Robot Selects, Picks Melons.
- *Purdue University Perspective*. Winter, 1993. Ag Robots May Replace Manual Farm Labor.
- *The Farmer's Exchange*. March 5, 1993. By 2000, Farmer's Choice to Be Robot or Pickup.
- *The Rural Shopper*. July 19, 1993. Ag Robot Harvests Fruits, Vegetables.
- *The Wall Street Journal*. June 9, 1993. p A10. Israel Moves to Automate Its Agriculture: Use of Robots Grows as Palestinian Problem Worsens.
- *Hoosier Banker*, October 1993. Ag Robot Harvests Fruits, Vegetables.
- *Davar Hashavua*. September 25, 1992. pp 20-21
- *Western Grower & Shipper*. August, 1992. pp 16-17. Automation Advancements
- *Forefront*. 4(1) Winter 1992. p 1. But Do They Do Windows?

APPENDICES

A. Field Data

Table A1. Field Observations Taken at Ein Tamar on April 21, 1993 (Beginning of Season)

Far Vis	Near Vis	#Loops	Near Reached	Picked	False Pick	Actual
3	1	3	1	1	2	1
3	2	1	2	0	1	2
2	2	1	1	2	0	2
3	4	1	1	2	2	1
5	5	1	5	4	1	4
4	4	1	4	3	1	3
5	3	1	2	3	3	2
1	1	1	1	1	0	1
6	2	2	2	2	4	2
2	3	1	3	3	0	2
1	2	1	2	2	1	1
3	2	1	1	2	0	3
4	2	1	2	2	2	2
2	2	1	2	2	0	3
2	1	1	3	3	1	1
1	2	1	2	1	0	1
2	2	2	3	3	1	1
4	1	2	4	4	2	2
1	2	1	2	1	0	1
2	1	1	1	1	1	1
4	1	2	1	2	1	1
1	1	1	1	1	0	1
1	2	2	2	1	1	1
1	1	1	2	1	1	1
3	1	1	1	1	2	1
4	2	1	3	3	1	3
4	2	1	1	1	2	2
2	2	1				1
4	1	3	3	3	1	3
2	2	1	2	2	0	2
2	1	1	2	2	0	2
2	1	1	2	2	0	2
2	3	1	3	3		2
6	5	2	4	4		5

Table A2. Field Observations Taken at Ein Tamar on April 28, 1993 (End of Season)

Far Vision	Near Vision	#Loops	Near Reached	Picked	False Pick	Actual
1	1	1	1	1		1
1	1	1	1	1		1
4	1	1	1	1	1	1
2	2	1	2	2	0	2
1	1	1	1	1	0	1
2	1	2	1	1	2	1
2	1	1	1	1	0	2
1	1	1	1	1	0	1
1	1	1	1	1	0	2
1	1	1	1	1	0	1
1	3	1	3	3	2	1
1	1	1	1	0	0	1
4	3	1	3	2	0	3
2	1	2	1	1	0	1
1	1	1	1	1	0	1
1	1	1	1	1	0	1
3	1	1	1	1	0	3
4	1	1	1	1	0	3
1	2	1	1	1	0	2
1	1	1	1	1	0	1
2	3	1	3	3	0	3
2	3	2	2	2	1	2
4	2	2	3	3	1	3
3	2	3	2	1	0	2
1	2	1	2	2	0	2
2	3	2	3	3	1	2
1	2	1	2	2	1	1
3	4	2	4	4	1	3
2	0	1				1
1	2	1	1	1	1	1
1	1	1	1	1	0	1
1	0	0	0	0	0	1
2	2	1	2	2	1	1
2	3	3	2	2	0	2
1	2	1	2	2	0	3
1	1	1	1	1	0	1
2	2	1	2	2	1	2
3	3	2	3	3	0	3
1	1	1	1	1	0	2
1	1	1	1	1	0	1
1	1	1	1	1	0	1

Far Vision	Near Vision	#Loops	Near Reached	Picked	False Pick	Actual
1	1	1	1	1	0	1
1	1	1	1	1	0	1
1	1	1	1	1	0	1
2	2	2	2	2	0	2
1	3	1	3	3	1	1
2	1	1	2	2	1	1
2	3	1	2	2	1	2
1	2	1	2	2	1	1
2	1	1	1	1	0	2
4	1	4	1	1	0	1
2	2	2	2	2	0	3
5	3	2	3	3	0	3
1	1	1	1	1	0	1
2	2	2	3	3	1	2
2	2	1	2	2	0	2
1	1	1	2	1	0	1
1	1	1	1	1	0	1
2	2	1	2	2	0	2
2	3	1	3	3	0	5
1	0	1	1	1	0	1
4	1	2	2	2	1	2

Table A3. Field Observations Taken at Yifat on July 12, 1993

Far Vis	Near Vis	#Loops	Near Reached	Picked	False Pick	Actual	2nd pick	Nitlash
1	1	1	1	1	0	1	1	
3	1	3	3	1		2	1	
1	1	1	1	1		1	1	
1	1	1	1	1		1	1	
2	1	2	2	1		2		
2	2	1	1	2		2	1	1
1	1	1	1	1		1	1	
2	1	2	2	1		2	1	
3	2	2	2	2		2	0	1
2	2	2	2	1		2	1	
2	1	1	1	1		1		
2	4	2	4	3	1	3	3	
3	2	2	2	2		2		
3	1	1			2	2		
3	2	1	1	2	2			
1	1	1	1	1	1			
1	1	1	1	1	1	1		
1	1	0	0	1	1			
1	1	1	1	1	1	1		
1	1	1	0	1	1			
1	1	0	1		1			
1	1	0	1		1			
2	1	1	2	2	2	2		
2	1	1	1	1	2			
1	1	1	1	1	0	1	0	1
1	1	1	1	1	1	1		
2	2	2	2	2	2	1		
1	1			1				
1	1	1	1	1	1	1		
1	1	1	1	1	1	1		
1	1	1	1	1	1			
1	1	1	1	1	1			
2	3	2	3	3	4			
2	1	1	1	1	2			
1	1	1	1	1	1			
2	1	2	1	1	2			
1	1	1	1	1	1			
1	1	1	1	1	1			
1	1	1	1	1	1			
1	1	1	1	1	1			
1	1	1	1	1	1	1		
4	3	1	3	3	3			
2	1	1	1	1	1			
1	1	1	1		1			

Far Vis	Near Vis	#Loops	Near Reached	Picked	False Pick	Actual	2nd pick	Nitlash
1	1	1	1	1	1	1		
1	1	1	1	1	1			
2	1	1	1	1	2			
2	3	2	3	3	1	2		
2	2	2	2	2	2			
1	1	1	1	1	1			
1	1	1	1	1	1			
1	1	1	1	1	1			
1	1	1	1	1	1	1		
2	1	1	2	2	2	1		
2	2	2	2	1	2	1		
1	1	1	1		1			
2	2	2	2	2	2			

Table A4. Field Observations Taken at Yifat on July 13, 1993

Far Vis	Near Vis	#Loops	Near Reached	Picked	False Detect	Actual	2nd Pick	Nitlash
5	5	3	4	4	5	1	1	
3	3	3	3	2	1	2		
3	2	2	1	2	3	1		
1	2	1	1	1	1	2		
3	2	2	2	1	2			
4	4	1	2	2	4			
3	3	3	3	4	4	1		
1	1	1	1	1	2			
2	2	2	2	2	2	1		
1	1	1	1	1	1	1		
3	3	2	2	3	2			
2	1	3	1	3	2	3		
1	1	1	1	1	1	1		
2	1	1	1	1	1			
1	1	1	1	1	1	1		
5	4	3	2	2	4	1		
2	2	2	1	1	2	1		
2	1	1	1	1	1	1		
1	1	1	1	1	1			
3	1	1	2	2	2	3		
2	2	1	2	2	1	2	2	
1	3	1	2	2	2	1		
4	2	2	2	3	4	1		
2	1	1	1	1	2			
2	2	2		2				
2	1	2	2		4			
2	5	1	2		3	1		
2	1	1	1	1	2			
1	1	1	1	1	0	1		
3	1	1	2	2	3			
4	1	3	3	3	4			
3	2	2	3	3	3			
4	1	3	1	1	1	2		
2	2	2	1	1	2			
2	1			2				
3	1	4	0	1	1			
2	2	1	0	2	1	2		
1	2	1	1	1	1			
2	2	1	2	1	3	1		
2	1	1	1	1	2			
3	2	2	4	3	1	3		