Localization of spherical fruits for robotic harvesting

Article *in* Machine Vision and Applications · November 2001

DOI: 10.1007/PL00013271 · Source: DBLP

2 authors:

Alessio Plebe
Università degli Studi di Messina
80 PUBLICATIONS 357 CITATIONS
SEE PROFILE

Giorgio Mario Grasso
Università degli Studi di Messina
48 PUBLICATIONS 203 CITATIONS
SEE PROFILE

Machine Vision and Applications (2001) 13: 70–79

Machine Vision and
Applications
© Springer-Verlag 2001

Localization of spherical fruits for robotic harvesting

Alessio Plebe, Giorgio Grasso

Department of Mathematics and Informatics, University of Catania, V.le Andrea Doria 6, 95125 Catania, Italy; e-mail: {alessio,ggrasso}@dmi.unict.it

Received: 16 April 2000 / Accepted: 19 December 2000

Abstract. The orange picking robot (OPR) is a project for developing a robot that is able to harvest oranges automatically. One of the key tasks in this robotic application is to identify the fruit and to measure its location in three dimensions. This should be performed using image processing techniques which must be sufficiently robust to cope with variations in lighting conditions and a changing environment. This paper describes the image processing system developed so far to guide automatic harvesting of oranges, which here has been integrated in the first complete full-scale prototype OPR.

Key words: Fruit harvesting – Color clustering – Stereo matching – Visual tracking

1 Introduction

In most developed countries harvesting accounts for the highest fraction of the total production cost of fresh fruits, and a machine capable of solving the harvesting problem at a reasonable cost would be welcome in the market. For this reason several projects are being carried out worldwide (Juste and Sevila 1991; Harrell 1987; Molto et al. 1992; Sarig 1993; Edan 1995; Kondo et al. 1995).

Fruit picking robots is an application that is strongly dependent on sensory information. Fruits are distributed within the canopy in a quite unstructured manner, and it would be essentially impossible to explicitly model or use any cue known a priori. On a larger scale, the layout of the orchards includes rows of trees which are of a broad range of sizes, and the limited spacing between rows hinders an overall sensing of the environment.

The main way of controlling an efficient outdoor harvesting robot is using reliable sensory-based environment analysis, and from the various types of sensory information available, vision is the most important because of the high resolution that is achievable (Tillett 1991; Tillett and Tillett 1995).

When implementing an image processing system able to control a real harvesting robot, several serious problems soon arise due to the following requirements:

1. Coping with the extreme dynamic range of the lighting that occurs in a scene looking at a tree canopy from any possible orientation of the camera with respect to the sun.
2. Identifying objects partially obscured by leaves, branches, or overlapped with other fruits, sometimes in large clusters, and with a fairly large range of color (yellow-green to red).
3. Completely localizing in 3-D space the fruit, despite uncertainty regarding its shape (due to eccentricity) and size.
4. Returning the correct spatial information when the limited working space between the trees requires the use of wide-angle cameras which lead to a distorted perspective.
5. To be fast enough for the real-time requirements of the robot.

Extracting the depth information from visual information is also an important factor. Although other alternatives could be applied in principle, such as moving towards a fruit whilst sensing the distance (using a range sensor) until just before reaching it, early knowledge of the positions of all the fruit in 3-D space produces a noticeable time saving when globally planning the robot trajectories.

The fruits of interest for the robot application are located at the outside of the canopy; since the mechanical arms are not able to work deep inside the canopy, the available visual information can result in missing the inner fruits. Moreover, a scene captured from the outside will always present several partially occluded fruits. Even a single leaf is enough to cover a large portion of a fruit. On the other hand, overlapped fruits might easily appear as a single one. Citrus are typical trees with a dense foliage, and quite often 70–80% of the fruits have half or more of their surface obscured.

Probably the greatest difficulty arises from the extreme variation in the lighting (Slaughter and Harrell 1989). Fruits like red apples and oranges apparently separate very well in

This work was supported in part by European Community ESPRIT Project 6715 CONNY (Robot control based on neural network systems).

Correspondence to: A. Plebe

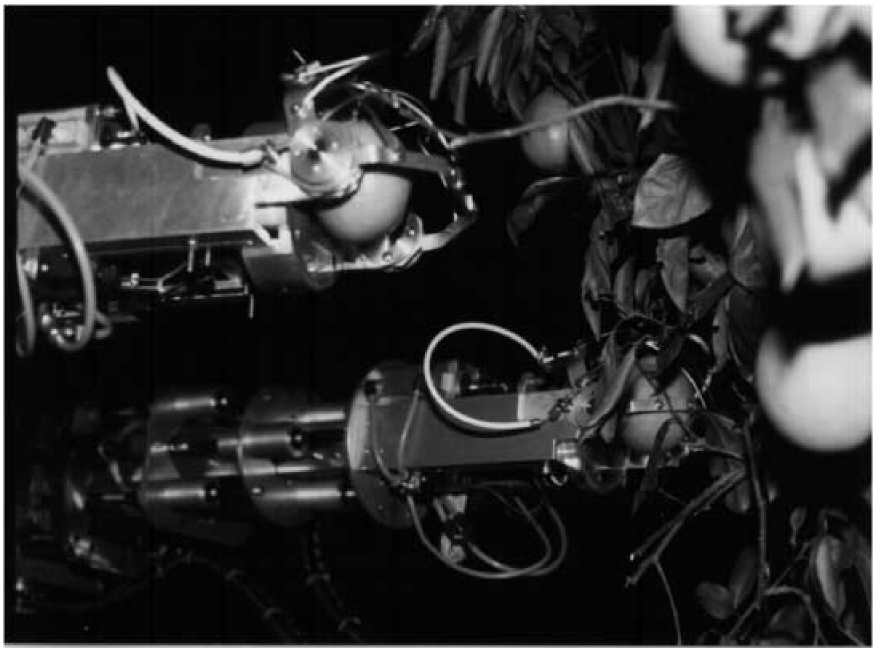


Fig. 1. Detail of the two telescopic electrical arms of the OPR in action

color space from a green background under controlled light conditions, but in an outdoor environment the differences in illumination are much greater than the chromatic separation of the fruit from the leaves. There is a technological upper limit determined by the dynamic range of the CCD of the camera; when irradiated by light with a very broad dynamic range, some areas may be saturated or some may appear black, or both types of area may occur, with the result that the sensed image is not suitable for further processing. Below this limit there are situations where the chromatic information is kept, but is weak and corrupted, and unfortunately such situations are quite common. The standard mechanisms for adapting the camera to different conditions, like automatic gain control and white-balance correction[1], which are indeed necessary for the robot to operate at all, are ineffective when extreme lighting differences occur within the same frame.

The system described here addressed the problems described above through a quite complex sequence of processing steps, which were aimed at providing a full localization of fruits in the scene for determining the robot motion. It was found that the system provided acceptable performance under real conditions.

Most of the process is customized for the mechanical characteristics of the OPR prototype. It is a tank-like vehicle with a five-degree-of-freedom hydraulically driven base that supports two three-degree-of-freedom electrically driven telescoping arms. The most important components of the total robot kinematic chain are the two electrical arms which perform fast-motion harvesting and carry a small high-resolution color camera in the center of the gripper. During harvesting the base of the arm is moved into a fixed position, then oranges within reach of the two arms are harvested. Details of the harvesting arm are shown in Fig. 1.

Since the motion-sensing cameras were positioned on the telescopic arms, the final representation of the detected centers is not referred to a coordinate system fixed on the robot, but referred to the platform carrying the arms. From pairs of target centers in the 2-D image plane, coming from the segmentation of a stereo pair of images, a spherical joint representation is constructed using a neural network nonlinear interpolator. The target centers used for this purpose are preliminary coupled in pairs by a stereo matching process. The image detection is completed by a tracking task, which follows the fruit regions within the camera views and runs during the motion of the arm.

In the next section a brief overview of the system will be given, followed by details of the vision processing in 2-D and in the 3-D space.

2 Description of the architecture

The image processing architecture of the OPR is made of several steps, sketched in Fig. 2. For a complete description of the robot see Recce et al. (1996).

In every harvesting position, the two arms each collect a pair of stereo images. Traditional two-camera stereo vision is not feasible due to the geometry of the robot links, that impose a fish-eye-lens effect (Shah and Aggarwal 1997), with resulting high distortion and large disparities between the two far-apart camera stereo images. Therefore, stereo pairs are taken with the same camera, and with the minimum necessary offset.

The first step is to detect orange centers in the four 2-D images. After this stage, stereo matching is performed on each image pair for the same arm (and camera), to associate corresponding centers to unique oranges in space.

Next, pairs of 2-D coordinates are mapped into 3-D space. However, since the 3-D coordinates of the oranges are only used to drive the robot links, our approach has been to directly map from 2-D pairs (4-D coordinates)

[1] White balance correction is performed at the beginning of the harvesting operation, or when a dramatic change in lighting condition is detected by the operator.

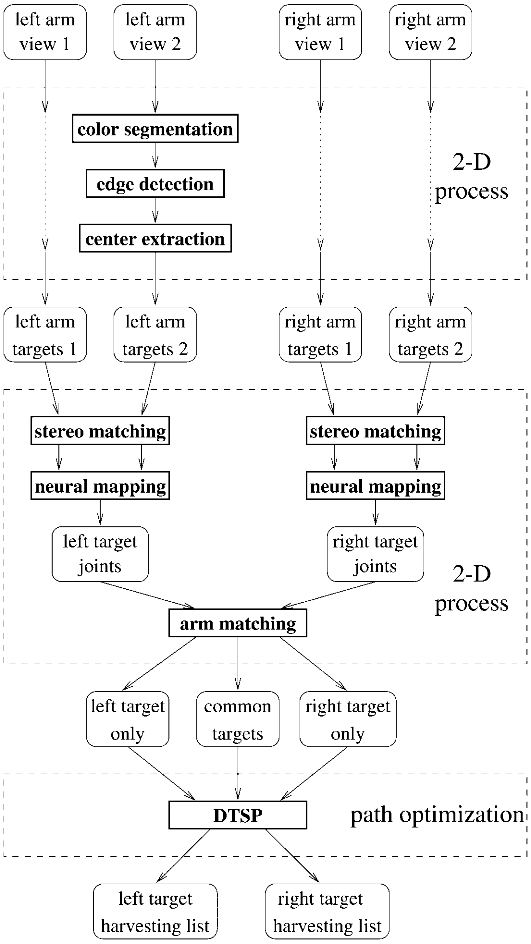


Fig. 2. Schematic of the vision processing architecture for the OPR

into link-joint coordinates (3-D for the telescopic spherical-coordinates arms).

The next step deals with the matching between oranges seen in both cameras, which may be picked by either one of the telescopic arms. To resolve this ambiguity and avoid the situation where both arms attempt to reach the same target, the two sets of orange centers represented in each arm-joint space are further transformed into a single virtual-joint-frame space that is placed in the middle of the origins of the two arms. Centers that belong to the overlapping working area of the arms (which is also the overlap area of the two cameras), are processed with another matching algorithm in order to maintain uniqueness.

The output of this step will be used by an optimization process, called DTSP (*double traveling salesman problem*), which computes the two full paths for the arms whilst at the same time attempting to minimize the overall path and the mechanical load difference between the two arms. This last algorithm will not be described here, since is not actually part of the visual processing (Plebe and Anile 2001).

Besides the complete orange-detection analysis, vision is also used during the actual harvesting task. Since the positions of the fruits may change in time from the initial scene used for the analysis, because of the unloading of branches or wind conditions, it is useful to have a visual refinement of the target position when approaching it. The tracking uses very fast center detection, based on the same color segmentation techniques of the initial scene analysis, but limited to a single-color (i.e., orange) process, supposing that when the arm is close to the target position, most of the image will be covered by one target orange.

3 2-D processing

This first step is probably the most critical, since errors at this stage may be reflected all the way through the processing chain.

In order to minimize the stereo matching problem, the 2-D process must minimize the number of false targets and detect over 75% of the fruit. Furthermore, the center of each orange must be found even if a large fraction of it is occluded.

3.1 Color clustering

Color segmentation is usually performed by mapping the three color components of each pixel (RGB; red, green, and blue) into the HSV (hue, saturation, and value) color space. Since a particular hue and saturation identifies the color orange, it is convenient to represent the colors in terms of just these two quantities (Gonzalez and Woods 1992). In Slaughter and Harrell (1987), the hue and saturation were used to identify individual oranges.

In our approach the color information contained in the images is represented in a modified HSV color space. The modified color space is constructed by projecting all of the pixels along a direction perpendicular to the normal HS plane (see Fig. 3b). This operation pushes pixels with a low saturation or low intensity close to the space center, and pixels – which are more significant – possessing a large product of saturation and intensity far from the center (Grasso and Recce 1997).

The coordinates of projected points are found by solving a system of equations that includes the equation of the projection plane ($x + y + z = 1$) and the lines of projection for an arbitrary pixel ($Gx - Ry = 0$; $By - Gz = 0$), where R, G, and B are the red, green, and blue values of a pixel. After solving the system of equations it is found that the coordinates x, y, and z are the normalized color values of R, G, and B.

In the parallel projection of pixels in RGB space onto the [111] plane, the solution to the system of equations is:

$$\begin{aligned} x &= \frac{1}{3}(1 + 2R - G - B) \\ y &= \frac{1}{3}(1 - R + 2G - B) \\ z &= \frac{1}{3}(1 - R - G + 2B) \end{aligned} \qquad (1)$$

The parameters x, y, and z describe a 2-D color space embedded in three dimensions. It is convenient to describe this color space with a 2-D representation (X,Y) in the plane of the projection.

$$X = \frac{\sqrt{3}}{2}(y - z) \qquad (2)$$

$$Y = x - \frac{1}{2}(y + z) \tag{3}$$

In the case of point projection, substituting the expressions for x, y, and z from (1) into the equations above, it is possible to write:

$$X = \frac{\sqrt{3}}{2}\frac{G - B}{R + G + B} \tag{4}$$

$$Y = \frac{R}{R + G + B} - \frac{1}{2}\frac{G + B}{R + G + B} \tag{5}$$

While, in the case of parallel projection:

$$X = \frac{\sqrt{3}}{2}(G - B) \tag{6}$$

$$Y = R - \frac{1}{2}(G + B) \tag{7}$$

Several observations can be made from the representations obtained from the two projection methods. In point projection, corresponding to the classical HS analysis, colors are normalized, so the quantity $V = R + G + B$ must be evaluated for each pixel. Secondly, point projection causes the pixel values to spread, so errors in low intensity pixels are amplified. Since most of the intrinsic causes of noise are independent of the intensity level of individual pixels, the relative amount of error is larger for low intensity pixels. In the method of projection used here, such low intensity pixels are close to the origin of the RGB space and are therefore close to the origin of resulting color space (X,Y). This new color space, unlike the traditional HS triangle, has a hexagonal shape (this can be verified by substituting the maximum values of R, G, and B in Eqs. 6 and 7). The distance from the center of this hexagon (see Fig. 3) is: $\Sigma = SI$, where S is the saturation and I is the intensity. Since S and I have values less that one, their product will be small if either one is small ($\ll$ 1). Because of this property, the pixels with a low product of saturation and intensity (i.e., big relative error) are found close to the hexagon center.

In Fig. 3, the increase in separation between the two regions of interest is clear. In the HS space (Fig. 3c), pixels are spread over a larger area and there is less distinction between orange and the background colors. In the HΣ space (Fig. 3d), two well-separated clouds of pixels can be distinguished, one in the region of red-orange, which corresponds to the distribution of oranges' colors, and the other crossing the origin horizontally from blue to green, which corresponds to the background. The overlap between the region of interest and the background in the HΣ color space is significantly reduced. This last point is of key importance, since if pixels belonging to the orange cluster are found within the background region or vice versa, the segmentation leads to either a loss of interesting regions or the inclusion of some of the background.

The classification of regions in the color space, and segmentation of the image of an orange tree, is performed using a multilayer perceptron, trained by the back-propagation or errors algorithm (Rumelhart et al. 1986; Benhanan et al. 1992). A database of 800 images taken in different lighting conditions is used to construct a training set. The training set includes pixels from a set of samples containing only oranges, and from a second set that contains only background fragments (e.g., tree branches, sky, and grass). After the back-propagation algorithm has converged, the network is used to build a look-up table of integer values for each possible point in the color space.

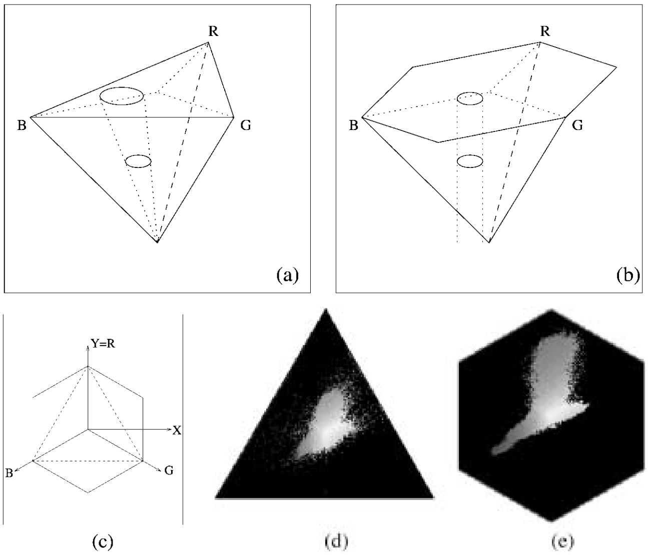


Fig. 3. a–c. Projection plane color-space: **a** point projection; **b** parallel projection; **c** the color HS triangle (*dashed*) and HΣ hexagon (*solid*). **d,e.** Distributions of colors for a typical orange tree image: **d** in the HS space derived from point projection; **e** in the HΣ space derived from parallel projection

3.2 Adaptive edge fitting

After the orange regions of the image are found, the background regions are set to black and the resulting image is smoothed (5 × 5 Gaussian convolution), thresholded, and the edges are detected.

Pixels on the edges are subsequently sorted to obtain a list of contiguous points. All edges are closed curves, and each closed curve is labeled as a separate object. The ordered lists of pixels in each object are used to calculate the curvature and the radius of curvature of a portion of the edge. The calculation is performed by solving the system of equations

$$\frac{c}{a}\theta = \sin\theta \qquad r = \frac{a}{2\theta} \tag{8}$$

where r is the radius of curvature, a is the length of the arc of edge, c is the length of the segment joining the starting and ending point of the arc, and θ is the half of the angle corresponding to the arc. The first equation has no analytic solutions, but it has one solution other than the zero when $c/a \in [0; \frac{2}{\pi}]$ and can be solved by a numerical algorithm.

The number of points in the portion of the edge used to calculate the curvature is adjusted recursively until the length of the arc is near to the value of the radius, within a specified tolerance.

The calculation of the radius is used to estimate the position of the center of curvature, and the result of this computation is stored. The algorithm repeats this calculation shifting one pixel at the time along the edges of each object.

When a sharp variation in the position of the estimated center is detected, the algorithm calculates the mean value

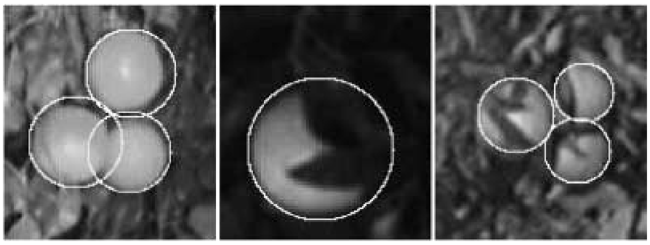


Fig. 4. Examples of regions from full scenes. The overlayed lines show the detected oranges: a cluster (*left image*), an occluded single orange (*center*), and an occluded cluster (*right image*)

of the positions of centers and of the radii for all guesses and begins to accumulate values for another orange. The mean value is calculated with a weighted sum, where the weight is proportional to the number of pixels used in the calculation of each guess.

This method has advantages over other, more conventional approaches, such as the Hough transform. The Hough transform requires an extra convolution (e.g., Sobel_x and Sobel_y, instead of just nondirectional edge detection) and is slower. Moreover, when applied to the orange detection problem the Hough transform must operate in three dimensions. This has been tested during the preliminary study of spherical fruit localization. A 3-D space containing information about the (X,Y) position of the center and the radius needed to be constructed, which had a very large computational overhead. Also, it was empirically found that the Hough transform does not work well with clusters of oranges, or when a substantial part of an orange's surface is occluded. For this reason the Hough approach was abandoned during the early development of the algorithm.

At the end of this processing a second pass is used to ensure that the results are consistent. This algorithm is used to prune the output of the previous steps, by eliminating all duplicate references to any single orange. The algorithm calculates the area of overlap of each pair of identified circular regions, and if the overlap is greater than a threshold (40% of the area) then the circle that accumulated the lowest score during the edge-fitting process is rejected. In addition, detected oranges with unreasonable sizes are discarded. Orange radius has been found experimentally to range between 25 mm and 55 mm, with a maximum eccentricity value of 1.3. In determining the range of allowed sizes, the scaling introduced by the distance is taken into account. The eccentricity value did not affect the performance of this algorithm.

3.3 Tracking

This process takes place during the final approach of the arm to the orange, and is a complete 2-D analysis whose outputs are simply the coordinates of one orange in image space. It is expected that a single orange will be in the field of view, or at least that one orange nearest to the center is larger than all the others.

The speed requirements of this process are much more stringent than for the global scene analysis, since vision is directly wired as feedback into the motion control loop. The color segmentation is exactly the same, since it is performed by hardware look-up tables. The rest of the process is different and reduced to the essential steps.

First, a preliminary examination of a small region around the initial image coordinates is used to check whether sufficient pixels are present in the image to confirm the location. More precisely, in a 5×5 square centered on the expected location, at least half of the pixels should be labelled as "orange". If not, a search is performed. Position checking attempts to determine the true center of the target orange by scanning the image in four directions outwards from the current position until the boundaries are detected. The four points found are used to fit a circle, and the center of this circle is used as a new estimate of the orange center. If the resulting estimate is within preset radius limits, the coordinates are returned, otherwise another search is performed.

When the tracking is not centered on a suitable orange, a rectangular "spiral" search is performed – initially with a small step size – followed, if necessary, by a larger step size until the boundary of the image is reached. Normally this search strategy finds the orange very quickly if the positional error is small (for example, if the orange is displaced by wind).

4 3-D processing

The 3-D localization starts from four images: the two pairs of views for each arm, as produced by the 2-D process described above.

4.1 Stereo matching

Matching is in general the difficult task in stereo vision. For the OPR the complexity is reduced, since the job is to map a few sparse points in space (i.e., orange centers) and not continuous objects. However, difficulties arise because of the clustering of oranges, which can lead to the apparent number of centers being less then the actual number of oranges. The opposite effect is due to small leaves and branches breaking the fruit contour and leading to the computed number of centers being too large. The method used here is based on classical epipolar constraints (Sonka et al. 1993), combined with a probabilistic relaxation process.

For each left target, disparities among both directions are computed in the right image for all targets:

$$\begin{aligned} \Delta X_{l,r} &= x_r - x_l \\ \Delta Y_{l,r} &= |y_r - y_l| \end{aligned} \qquad (9)$$

where $l = 0, \ldots, L-1$ and $r = 0, \ldots, R-1$, with L and R being the number of targets in the left and right images, respectively. Since the two views are generated by a horizontal shift of the robotic arm, only positive values of $\Delta X_{l,r}$ in (9) will be physically possible.

All possible matches are selected on the basis of disparity constraints:

$$\begin{aligned} \Delta X_0 \le \Delta X &\le \Delta X_M \\ \Delta Y &\le \Delta Y_M \end{aligned} \qquad (10)$$

ΔX_0 and ΔX_M are the physical limits of the workspace, being the farthest and closest reachable distances, respectively; ΔY_M is the largest vertical mismatch due to the lens effects, as experimentally determined.

For all matches satisfying the (10) constraints, the ΔX disparity is stored in a $L \times R$ matrix $\Delta\mathbf{X}$. In general, there will be more then one element on each row and column that are multiple matches, and the purpose of the algorithm is to reduce to the most probable single matches. In the relaxation process, at each step n the mean disparity $\overline{\Delta X}_n$ is computed:

$$\overline{\Delta X}_n = \mathrm{E}(\Delta\mathbf{X}_n)$$

Then, elements along the column and the row are selected according to the smallest deviation σ from $\overline{\Delta X}$:

$$\sigma_{ij} = (\Delta X_{ij} - \overline{\Delta X})^2 \qquad (11)$$

A new matrix $\Delta\mathbf{X}$ is generated as a logical OR operation between two matrices:

$$\Delta\mathbf{X}_{n+1} = \left(\begin{bmatrix} c_{11} & \cdots & c_{1r} \\ \vdots & \ddots & \vdots \\ c_{l1} & \cdots & c_{lr} \end{bmatrix}_n \vee \begin{bmatrix} r_{11} & \cdots & r_{1r} \\ \vdots & \ddots & \vdots \\ r_{l1} & \cdots & r_{lr} \end{bmatrix}_n \right) \Delta\mathbf{X}_n \qquad (12)$$

where the elements c_{ij} and r_{ij} are:

$$c_{ij} = \begin{cases} 1 & \text{if } \sigma_{ij} = \min\{\sigma_{ik}; k = 0, \ldots, r\} \\ 0 & \text{otherwise} \end{cases} \qquad (13)$$

$$r_{ij} = \begin{cases} 1 & \text{if } \sigma_{ij} = \min\{\sigma_{kj}; k = 0, \ldots, l\} \\ 0 & \text{otherwise} \end{cases}$$

This step is based on a simple consideration. The global epipolar constraints in (10) are based on any possible geometry of the tree compatible with the robot kinematics, and the process therefore performs a broad filtering of the candidate matches in (9). On the contrary, any single scene will be just a group of branches or a portion of the canopy; therefore it is unlikely to have fruits scattered over the whole range of depths, and minimizing σ in (11) is an effective criteria for reducing multiple matches.

A new mean disparity $\overline{\Delta X}_{n+1}$ is then computed again from $\Delta\mathbf{X}_{n+1}$, and the process is iterated until either one of the following conditions occurs:

$$\begin{bmatrix} c_{11} & \cdots & c_{1r} \\ \vdots & \ddots & \vdots \\ c_{l1} & \cdots & c_{lr} \end{bmatrix}_n = \begin{bmatrix} r_{11} & \cdots & r_{1r} \\ \vdots & \ddots & \vdots \\ r_{l1} & \cdots & r_{lr} \end{bmatrix}_n \qquad (14)$$

$$\left|\overline{\Delta X}_{n+1} - \overline{\Delta X}_n\right| < \epsilon \qquad (15)$$

where c and r are defined in (10) and ϵ is a small positive number. The condition (14) means that there exists a set of single matches $\mathscr{M}$:

$$\mathscr{M} \equiv \{\{l, r\} \;:\; \Delta X_{lr} \not= 0\} \qquad (16)$$

with $|\mathscr{M}| \leq \min(L, R)$. If (15) occurs first, there will be multiple matches left in $\Delta\mathbf{X}$; these will be eliminated by simple heuristics, which takes out of a row or column the element with a value closest to the average $\overline{\Delta X}_n$.

Figure 6 shows two scenes, and the corresponding initial $\Delta\mathbf{X}$, with $L = 19$, $R = 15$, and $\overline{\Delta X} = 97$. In just two iterations condition (14) is met, with a resulting set of 14 target matches:

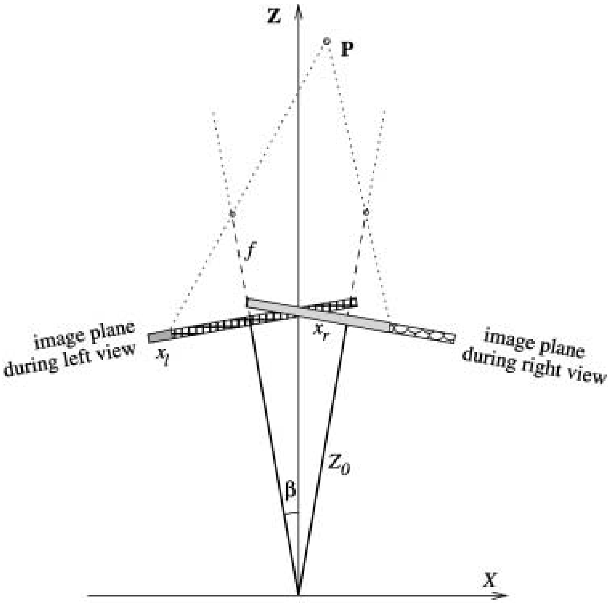


Fig. 5. Schematic of divergent cameras producing stereo vision

$$\mathscr{M} = \big\{\{3, 0\}\ \{5, 1\}\ \{6, 2\}\ \{7, 3\}\ \{8, 4\}\ \{9, 5\}\ \{10, 6\}\ \{11, 7\}\ \{13, 8\}\ \{14, 9\}\ \{15, 10\}\ \{16, 11\}\ \{17, 12\}\ \{18, 13\}\big\}$$

4.2 Mapping in joint space

The geometry of the stereo vision in this study is different from the standard method that uses twin cameras with parallel image planes. In this case, the two image positions are generated by a rotation along the direction normal to the vector joining the image plane with the robot link reference frame, at the origin of the latter.

The vector is rotated by an angle $\pm\beta$ away from the direction of view of the camera, as sketched in Fig. 5.

With this scheme Z is no longer only a function of the horizontal disparity. Nevertheless, depth information is still a function of only horizontal information, and can be derived analytically from the following equation system:

$$x_l = f\,\frac{X\cos\beta + Z\sin\beta}{f + Z_0 - Z\cos\beta + X\sin\beta} \qquad (17)$$

$$x_r = f\,\frac{X\cos\beta - Z\sin\beta}{f + Z_0 - Z\cos\beta - X\sin\beta}$$

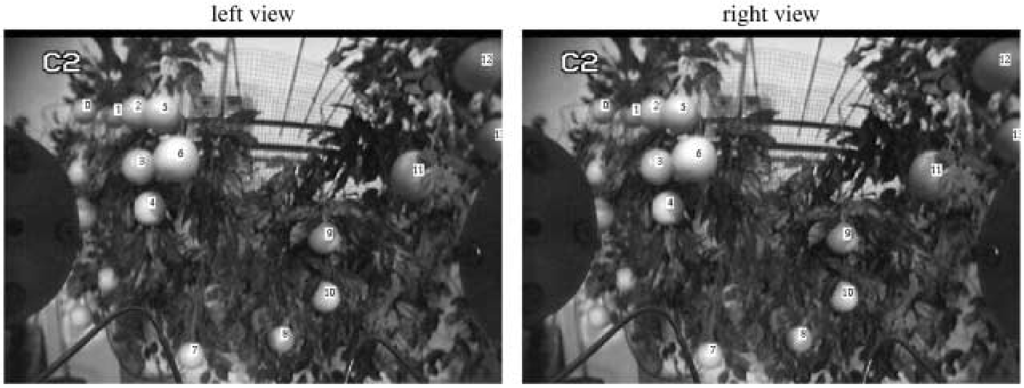
Solving for Z:

$$Z = (f + Z_0)\,\frac{x_r\,A - x_l\,B}{C - D} \qquad (18)$$

where f is the focal length of the cameras and with the following positions:

$$A = (f\ \cos\beta - x_l\ \sin\beta)$$
$$B = (f\ \cos\beta + x_r\ \sin\beta)$$
$$C = (f\ \sin\beta - x_r\ \cos\beta)\ (x_l\ \sin\beta - f\ \cos\beta)$$
$$D = (x_l\ \cos\beta + f\ \sin\beta)\ (f\ \cos\beta + x_r\ \sin\beta)$$

The application of (18) to a real camera requires the use of local corrections due to the distortions introduced by the



$$
\begin{array}{c|ccccccccccccccc}
 & 0_r & 1_r & 2_r & 3_r & 4_r & 5_r & 6_r & 7_r & 8_r & 9_r & 10_r & 11_r & 12_r & 13_r & 14_r \\
\hline
0_l & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1_l & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
2_l & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
3_l & 87 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
4_l & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
5_l & 130 & 89 & 69 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
6_l & 0 & 117 & 97 & 0 & 0 & 65 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
7_l & 0 & 0 & 0 & 99 & 0 & 0 & 61 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
8_l & 0 & 0 & 0 & 0 & 94 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
9_l & 0 & 0 & 134 & 0 & 0 & 102 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
10_l & 0 & 0 & 0 & 0 & 0 & 0 & 115 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
11_l & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 86 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
12_l & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
13_l & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 87 & 0 & 0 & 0 & 0 & 0 & 0 \\
14_l & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 98 & 0 & 0 & 0 & 0 & 0 \\
15_l & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 96 & 0 & 0 & 0 & 0 \\
16_l & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 120 & 0 & 0 & 0 \\
17_l & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 113 & 0 & 0 \\
18_l & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 100 & 0
\end{array}
$$

Fig. 6. Matching between two stereo images and the corresponding disparity matrix

system of lenses which is, in the case of very-wide-angle optics, particularly complex.

To overcome this problem, a neural network approach has been adopted to learn the mapping directly from the pair of 2-D coordinates into joint coordinates. This has the advantage of incorporating in a single step nonlinear lens distortion and the inverse kinematics of the mechanical system.

The network performs the following mapping function:

$$[\mathbf{I}_l, \mathbf{I}_r] \;\rightarrow\; \mathbf{J} \tag{19}$$

where $\mathbf{J} = \begin{bmatrix} \theta \\ \alpha \\ r \end{bmatrix}$ is the vector of the three joint coordinates, as indicated in Fig. 7, where both arms are depicted and referenced as arm A and arm B, and $\mathbf{I}_{l,r} = \begin{bmatrix} x \\ y \end{bmatrix}$ is the vector of the target position in the image plane for a single target.

The neural network is a multilayer network with supervised back-propagation learning; the training set has been built by covering with regular patterns the complete working space of the robot. The input to the network is generated by processing the two images taken at the predefined stereo camera positions, and the training output is generated by manually moving the end-effector on each target whilst taking the corresponding encoder readings. The network is organized with 4 input and 3 output neurons according to (19); experiments have shown that using 8 neurons for the hidden layer is optimal.

4.3 Matching between arm position and final output

The result of the previous step performed for all of the oranges in the scenes from the left and right arms is two sets of joint vectors **J**:

$$\mathscr{A} = \{\mathbf{J}_a\}, \quad \mathscr{B} = \{\mathbf{J}_b\}$$

where a and b refer to the left and right elements, respectively. Due to the overlap between the working areas of the two arms, elements of $\mathscr{A}$ and $\mathscr{B}$ may refer to the same target orange, and it is necessary to identify all common targets.

For this it is useful to refer joint vectors to a common virtual Cartesian coordinate system, integrated with the hub of the left arm, as shown in Fig. 7. The transformation will produce the two sets

$$\mathscr{A}^{XYZ} = \{\mathbf{J}_a^{XYZ}\}, \quad \mathscr{B}^{XYZ} = \{\mathbf{J}_b^{XYZ}\}$$

using the simple geometrical transformations from polar to Cartesian coordinate systems:

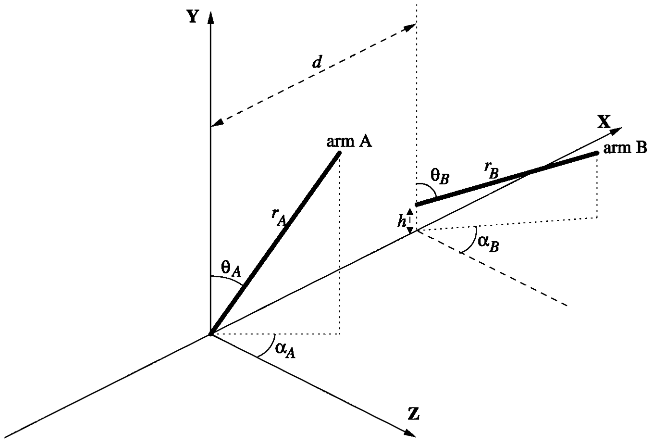


Fig. 7. The coordinate systems of the target positions for the two arms. d is the horizontal distance between the arms, and h is the vertical displacement between the hubs (usually very small)

$$\mathbf{J}_a^{XYZ} = \begin{bmatrix} r_a \, \sin\theta_a \, \sin\alpha_a \\ r_a \, \cos\theta_a \\ r_a \, \cos\theta_a \, \sin\alpha_a \end{bmatrix},$$

$$\mathbf{J}_b^{XYZ} = \begin{bmatrix} r_b \, \sin\theta_b \, \sin\alpha_b + d \\ r_b \, \cos\theta_b + h \\ r_b \, \cos\theta_b \, \sin\alpha_b \end{bmatrix}$$

The matching is performed using the same algorithm as used in the stereo matching. Matching between arms is less critical thanks to the availability of a third disparity along the Z direction.

As a result the original sets are reduced to three disjoint sets of target centers, $\mathscr{A}_{\mathscr{U}}{}^{XYZ}$, $\mathscr{C}_{\mathscr{U}}{}^{XYZ}$, and $\mathscr{B}_{\mathscr{U}}{}^{XYZ}$, where the first has a target which can be harvested by the left arm only, the latter targets are harvested by the right arm only, and $\mathscr{C}_{\mathscr{U}}{}^{XYZ}$ contains common targets. The following property hold for the sets:

$$\begin{aligned} &\mathscr{A}_{\mathscr{U}}{}^{XYZ} \bigcup \mathscr{C}_{\mathscr{U}}{}^{XYZ} \bigcup \mathscr{B}_{\mathscr{U}}{}^{XYZ} \\ &\quad \subseteq \mathscr{A}^{XYZ} \bigcup \mathscr{B}^{XYZ} \end{aligned} \tag{20}$$

The three sets are further processed by the DTSP algorithm to compute the optimal path for the two arms – distributing the elements of $\mathscr{C}_{\mathscr{U}}{}^{XYZ}$ to the left and right arms – producing the two ordered sets $\mathscr{A}_{\mathscr{P}}{}^{XYZ}$ and $\mathscr{B}_{\mathscr{P}}{}^{XYZ}$ with

$$\begin{aligned} &\mathscr{A}_{\mathscr{P}}{}^{XYZ} \bigcup \mathscr{B}_{\mathscr{P}}{}^{XYZ} \\ &\quad = \mathscr{A}_{\mathscr{U}}{}^{XYZ} \bigcup \mathscr{C}_{\mathscr{U}}{}^{XYZ} \bigcup \mathscr{B}_{\mathscr{U}}{}^{XYZ} . \end{aligned}$$

The software keeps track of the index of each vector during the above process, so that the final ordering is immediately imposed on the original sets $\mathscr{A}$ and $\mathscr{B}$, which are then used to by the motion controller for the mechanical harvesting task.

5 Results

It should be noted that it is difficult to properly evaluate the parts of the harvesting algorithm individually, since several parts of the system can be used to compensate for each other. For example, an error in the color segmentation of a portion of an orange might not cause an error in locating the fruit, since the remainder of the orange may provide sufficient information. Also, an error in the location of the orange can be corrected during the final approach by the tracking algorithm, or a falsely detected orange can be resolved during the stereo matching algorithm.

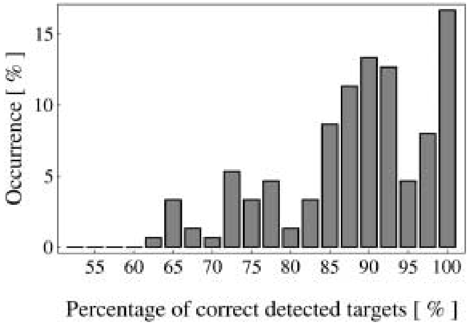


Fig. 8. Distribution of target detection success over 150 scenes. On the abscissa, the success of detection is expressed as the percentage of correct targets with respect to the total number of fruits within a scene. The ordinate shows the percentage of scenes for which a certain success rate (in a ±0.833% range) occurred. For example, in 13% of the scenes the success rate was about 90% (more precisely, between 89.17% and 90.83%)

Table 1. Results of orange detection in 50 scenes

	Oranges	Correct	False positive	False negative
Total	673	584	103	35
Mean per image	13.4	11.6	2.1	0.7
Percentage	100%	87%	15%	5%

When the robot is tested in real conditions, using its own real-time hardware, it lacks the external computing power that allows detailed monitoring and debugging of the intermediate processes. Therefore tests have been performed outdoors in real conditions for evaluation purposes, and laboratory tests and simulation of the algorithms on real images have been carried out for a better analysis of the results.

The complete processing task is depicted in Fig. 9.

5.1 Results of 2-D analysis

In order to evaluate the performance of the 2-D processing software, detailed statistics has been obtained from over 50 images containing 673 oranges. These images were collected under a range of lighting conditions and with different amounts of leaves and oranges in the images. The evaluation is based on the number of false positives and false negatives in the detection of oranges in an image. A false positive occurs when a something (e.g., a leaf) is considered an orange by the algorithm, or when the algorithm treats a single orange as more than one. A false negative occurs when a real orange in the image is not included, which often occurs when a cluster of two oranges is considered a single orange.

The overall performance of the algorithm is reported in Table 1. Looking in detail at the failures that occurred in the 2-D processing confirmed the failure mechanisms described in Sect. 1. For example, the segmentation algorithm

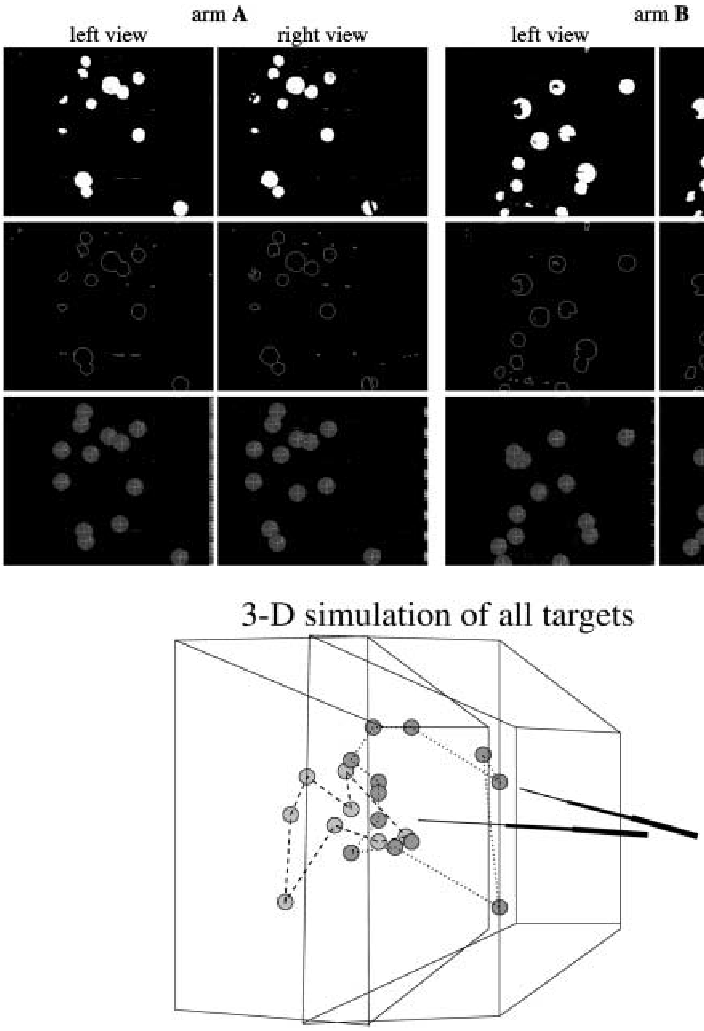


Fig. 9. The vision processing chain applied to the set of four images (from *left* to *right*): view 1 and 2 of the left arm, and view 1 and 2 of the right arm. The *top row* is the result of the color segmentation, followed by the edge detection, and the center detection. The *bottom figure* is the simulated 3-D location of the targets, with the two harvesting paths

had trouble with oranges that were either very brightly lit or very poorly lit, and with leaves that were brightly lit.

The orange detection part of the algorithm had significant problems with some types of occlusion. For example, leaves were more of a problem than clusters of oranges. The correct-detection statistics from over 150 scenes are shown in Fig. 8

5.2 Results of outdoor harvesting

The two environmental parameters which most affect the harvesting are the lighting conditions and the wind speed. The latter produces a disturbance that is proportional to its intensity up to a limit, above which harvesting becomes infeasible. Wind has two effects: one is to change the scene between the two snapshots used for the stereo mapping, altering the disparity and causing an error in the localization in joint space; the second effect occurs during the tracking phase, when the frequency of orange oscillation approaches the boundary of the system's response. In this condition the correcting movements of the end effector of the robot are not fast enough to follow the motion of the fruit. Lighting has already been mentioned as the condition that most influences the entire process.

Table 2. Results of outdoor harvesting for a total of 721 oranges

External conditions	% of hits	Average time per orange
Cloudy, no wind	85%	5.5
Clear sky, no wind	80%	5.3
Cloudy, windy	65%	9.7
Low sun angle, no wind	52%	7.5

Table 2 shows the overall harvesting score, accumulated over experiments that are classified according to the external conditions.

6 Conclusions

An image processing system for guiding an orange picking robot has been presented. Several aspects of data acquisition, image processing, 3-D vision, and robotic control have been analyzed in detail during the development of the automatic harvesting system, keeping in mind the difficult requirements of a real application. A new color space is used to segment images and extract the relevant information before the localization process. An adaptive edge-tracking algorithm has been adopted to extract orange centers, which proved more effective than traditional shape recognition methods where fruits occurred in clusters. A stereo matching procedure allows pairing of corresponding oranges in the two camera views, by feeding the position information to a neural network which has been trained to compute directly the coordinates in the arm-joint space.

Although data gathered during real experiments cannot fully validate the vision system, since they are affected by the performance of all the hardware components of the robot, the results achieved here are promising. Tests in controlled lighting conditions have shown the reliability of the system when operated within the dynamic range of the adopted optical system.

The construction of a complete commercial robotic system for harvesting oranges is still a far-reaching goal, mainly because of the additional requirements of reliability and low cost. However, the authors believe that the work presented here represents a proof of concept of the validity of automatic harvesting. In particular, the use of a full-scale robotic system in outdoor environments has demonstrated that the realization of an automated orange picking system is possible and could compare favorably with manual harvesting. A full-scale prototype harvesting robot is currently under further development to address the limitations described in this paper.

References

Benhanan U, Peleg K, Gutman P (1992) Classification of fruits by a Boltzmann perceptron neural network. Automatica 28: 961–968

Edan Y (1995) Design of an autonomous agricultural robot. Appl Intell 5: 41–50

Gonzalez R, Woods R (1992) Digital image processing. Addison-Wesley, Reading, Mass.

Grasso G, Recce M (1997) Scene analysis for an orange harvesting robot. Artif Intell Appl 11: 9–15

Harrell R (1987) Economic-analysis of robotic citrus harvesting in Florida. Trans ASAE 30: 298–304

Juste F, Sevila F (1991) Citrus: a European project to study the robotic harvesting of oranges. In: Proceedings of Second Workshop on Robotics in Agriculture and the Food Industry, Genova, Italy, 17–18 June, pp 187–195

Kondo M, Monta M, Fujiura T (1995) Fruit harvesting robots in Japan. Adv Space Res 18: 181–184

Molto E, Pla F, Juste F (1992) Vision systems for the location of citrus-fruit in a tree canopy. J Agric Eng Res 52: 101–110

Plebe A, Anile A (2001) A neural network based approach to the double traveling salesman problem. Neural Comput (in press)

Recce M, Taylor J, Plebe A, Tropiano G (1996) Vision and neural control for an orange harvesting robot. In: Proceedings of International Workshop on Neural Networks for Identification, Control, Robotics and Signal/Image Processing, 21–23 August, Venice, Italy

Rumelhart D, Hinton G, Williams R (1986) Learning internal representations by back-propagating errors. Nature 323: 533–536

Sarig Y (1993) Robotics of fruit harvesting – a state-of-the-art review. J Agric Eng Res 54: 265–280

Shah S, Aggarwal J (1997). Mobile robot navigation and scene modeling using stereo fish-eye lens system. Mach Vis Appl 10: 159–173

Slaughter D, Harrell R (1987) Color vision in robotic fruit harvesting. Trans ASAE 30: 1133–1148

Slaughter D, Harrell R (1989) Discriminating fruit for robotic harvest using color in natural outdoor scenes. Trans ASAE 32: 757–763

Sonka M, Hlavac V, Boyle R (1993) Image process anal mach vis. Chapman and Hall, Cambridge

Tillett R (1991) Image-analysis for agricultural processes – a review of potential opportunities. J Agric Eng Res 50: 247–258

Tillett N, He W, Tillett R (1995) Development of a vision-guided robot manipulator for packing horticultural produce. J Agric Eng Res 61: 145–154

Alessio Plebe was born in Casale Monferrato, Italy, in 1956. He graduated in electronic engineering from the University of Rome in 1981. He worked in the field of image processing at the Research Center of Agriculture Industrial Development SpA from 1982 to 1990. Since 1992 he has been the Head of the Robotics and Automation Department at the Consorzio per la Ricerca in Agricoltura nel Mezzogiorno (CRAM). He is currently with the Department of Mathematics and Informatics of the University of Catania.

Giorgio Grasso received a BS degree in applied physics from the University of Catania in 1993. He received a PhD degree in computer science at the New Jersey Institute of Technology in 2000. He is currently with the Department of Mathematics and Informatics of the University of Catania. His research interests include robotics, neural networks, image processing, and biomechanics.