Computer Vision Applied to Flower, Fruit and Vegetable Processing

Luis Gracia, Carlos Perez-Vidal, Carlos Gracia

Abstract—This paper presents the theoretical background and the real implementation of an automated computer system to introduce machine vision in flower, fruit and vegetable processing for recollection, cutting, packaging, classification, or fumigation tasks. The considerations and implementation issues presented in this work can be applied to a wide range of varieties of flowers, fruits and vegetables, although some of them are especially relevant due to the great amount of units that are manipulated and processed each year over the world. The computer vision algorithms developed in this work are shown in detail, and can be easily extended to other applications. A special attention is given to the electromagnetic compatibility in order to avoid noisy images. Furthermore, real experimentation has been carried out in order to validate the developed application. In particular, the tests show that the method has good robustness and high success percentage in the object characterization.

Keywords—Image processing, Vision system, Automation

I. INTRODUCTION

APPROXIMATELY, 900 million tons of vegetables and 500 million tons of fruits were produced in 2005 all over the world. The vegetable production is persistently increasing and the same trade is observed for fruit production. Fruit production grown about an 8% yearly in 2006, 2007 and 2008; this value is quite smaller for vegetables, around 3-4% for the same period. The European Union, Northern America and Japan are the major fruit and vegetable importers, mainly from South America, Russia and Asian countries [1]. At the same time, the flower industry is blossoming all over the world. Despite a sluggish global economy, estimates are that the sales of floral products increased significantly in the first years of the new century. According to Dr. Terri W. Starman (Department of Horticultural Sciences at Texas A&M University), in 2000 the wholesale value of the global flower market came to US$24.68 billion, which translated to a retail value of US$77 billion. Some types of flowers/fruits/vegetables (see Fig. 1) are especially relevant due to the great amount of units that are manipulated and processed each year over the world [2]. In fact, there are many works related with citrus [3] or other more unusual fruits like

L. G. Author is with the Universidad Politécnica de Valencia, Camino de Vera s/n, 46022 Valencia, SPAIN (e-mail: luigraca@isa.upv.es).

C. P.-V. Author is with the Universidad Miguel Hernández, Avda. de la Universidad s/n, Quorum V Building, 03202 Elche, SPAIN (e-mail: carlos.perez@umh.es).

C. G. Author is with the Universidad Politécnica de Valencia, Camino de Vera s/n, 46022 Valencia, SPAIN.

pomegranate [4].

(a)   (b)

(c)   (d)   (e)

Fig. 1 Many types of flowers/fruits/vegetables can be used with the application of this work: (a) rose flower (Rosaceae); (b) saffron flower (Crocus sativus); (c) sweet orange (Citrus sinensis); (d) tomato (Solanum lycopersicum); (e) artichoke (Cynara scolymus)

Beside this, computer or machine vision is becoming more and more used in industry, including agriculture, although in this field it is not enough widespread despite that the world agriculture production is increasing every. This paper presents the theoretical background and real implementation of an automated computer system to introduce computer vision [5] in flower, fruit and vegetable processing for recollection, cutting, packaging, etc. Some basic issues related to image processing are presented by [6] in order to increase machine vision in agriculture automation [7]. This research tackles some technical aspects that are important for automated machine vision [8], such as the camera, the acquisition card, the illumination, the electromagnetic compatibility (filtering and electric connections), etc.

The final target of the developed application is to obtain the position of the flower/fruit/vegetable centre (eliminating the background, peduncle and other non-relevant parts like sheets) and for example to compute the cutting point that allows to separate the element from its peduncle. This information (centre of the element) can be used for several types of agricultural tasks, such as recollection, cutting,
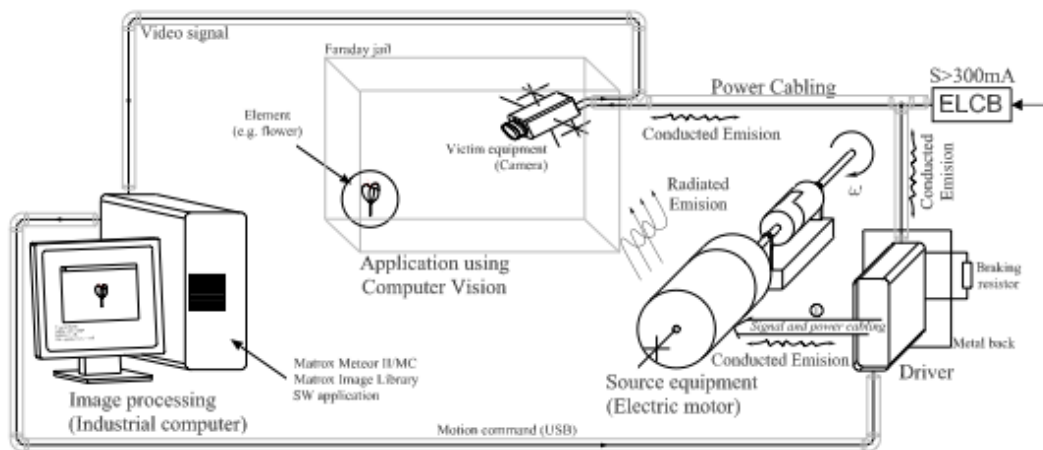
Fig. 2 Configuration setup of the system: camera; industrial computer for image processing; electric motor for a specific movement action; and electric connections

packaging, classification (by color or size), fumigation, etc., that can be performed by a robots [9]. The developed application is tested with different types of flowers, fruits and vegetables to check the effectiveness and robustness of the solution. An important feature of the proposed algorithm is that it is able to work with several elements inside the camera field of view. This adds flexibility to the application in order to work in the country or in a greenhouse, where the elements are very close to each other and the location of all them must be obtained in real time.

The specific objectives of this work are: (1) to describe issues related to computer vision for flower/fruit/vegetable processing; (2) to present the electromagnetic compatibility (EMC) considerations to take into account; (3) to show the final application developed and.

II. CONFIGURATION SETUP

Fig. 2 presents the configuration setup used in this work. The camera, which is inside a Faraday jail, acquires the image of the real flowers/fruits/vegetables. The video signal is sent from the camera to the industrial computer, which is equipped with a frame grabber that captures the image. The computer obtains the position of the centre of the detected objects and this information is used to process the element. Using this object characterization, the computer sends a "Motion command" (depending on the desired task: recollection, cutting, packaging, classification, etc.) through a USB (universal serial bus) link to the "Driver" that controls the "Electric motor" devoted to modify or manipulate the process.

The configuration setup shown in Fig. 2 has a significant problem of EMI due to the proximity of the "source" (electric drive and motor) and "victim" (vision system) of electrical noise. The effects of conducted and radiated interference are represented in Fig. 2. Therefore, all the issues described in next section "Electromagnetic compatibility" have been considered to obtain a proper performance of all the elements in the system. The electromagnetic compatibility directives

that are applicable to this work and to similar ones are: EN 50081-1 (residential & light industry) and EN 50081-2 (heavy industry) for Emission Standards and; EN 50082-1 (residential & light industry) and EN 50082-2 (heavy industry) for Immunity Standards and are summarized in next section.

A. Computer Features

Hardware: Industrial computer (PC type) for rack mounting; rack type keyboard and mouse; TFT color Display of 15" XGA (1024 x 768); Pentium®4 1,7GHz; 1GB of RAM; 4 serial ports RS-232; 3 USB ports; Hard Disk of 250GB; 3 slot for PCI.

Software: Microsoft Windows XP Professional with Service Pack 3; Matrox Image Library (MIL) 8.0; Microsoft Visual C++ 6.0 (programming environment); and Deep Freeze Standard 6.52 software (by Faronics®) to protect the core operating system and configuration files restoring the computer back to its original configuration each time it restarts.

B. Vision System Features

Camera (SONY XC-HR50): Monochrome; 1/3 type progressive scan (better than interlaced scans for capturing moving objects) CCD with square pixels; VGA resolution (704x495 pixels) capturing at a speed of 60 full frames/sec.; minimum illumination of 1.0 lux at F1.4; signal/noise ratio of 58 dB; electronic shutter from 1/100 to 1/30,000 sec.

Lens: Special for machine vision with C-mount; focal length of 12 mm; maximum aperture ratio of 1:1.4; iris range from 1.4 to 16.

Frame grabber (Matrox Meteor II/MC): captures from interlaced/progressive scan component RGB and single/dual channel monochrome analog video sources; sampling rates up to 30 MHz; three 256x8-bit LUTs connect two RGB or six monochrome video sources; trigger input and timer outputs; 32-bit/33 MHz PCI bus-master; real-time transfer to system or VGA memory.

Fig. 3 Illumination testbed designed in the laboratory with fluorescent lamps, electronic ballasts and voltage regulators. The structure is closed to avoid external perturbation

C. Illumination System

Image processing is very sensitive to illumination, and therefore constant environment conditions are very important to achieve a robust performance of the algorithm. An external enclosure has been used in this work to protect both the camera and the objects from the natural (or external) light. Therefore, inside the enclosure the illumination is controlled with fluorescent lamps (28W) with electronic ballasts and voltage regulators (model 75317-39 of Simon Electric). Electronic ballasts provide, in contrast to electronic starters, flicker-free illumination and high reliability. Fig. 3 shows several photographs of the illumination testbed designed in the laboratory.

III. ELECTROMAGNETIC COMPATIBILITY

A. Introduction

Typically, the vision system works in a noisy environment, due to the proximity of other machines and elements that introduce significant disturbances. Therefore, some kind of signal conditioning has to be considered to avoid data corruption. In this sense, Electromagnetic Compatibility (EMC) is the branch of electrical sciences that studies the unintentional generation, propagation and reception of electromagnetic energy with reference to the unwanted effects that such energy may induce, i.e. the Electromagnetic Interference (EMI). The goal of EMC is the correct operation of different electromagnetic equipment avoiding any interference effects. In order to achieve this, EMC pursues two different kinds of issues. Emission issues are related to the unwanted generation of electromagnetic energy by some source, and to the countermeasures which should be taken in order to reduce such generation and to avoid the escape of any remaining energies into the external environment. Immunity issues refer to the correct operation of electrical equipment, referred to as the victim, in the presence of unplanned electromagnetic disturbances. Interference mitigation is achieved by addressing both emission and immunity issues and can be tackled via hardware or software, although a combination of both approaches is usually the best solution. Subsections below describe the minimum measures required to allow the properly working of the camera and an electric motor and driver in the same application, as shown in Fig. 2.

B. Hardware Filtering and Connections

The first action to reduce the interference is to properly connect all the elements used in the application. The equipment's manufacturer provides important instructions about earth and ground connections that must be followed. In this sense, shield-to-ground connections have to be used. Moreover, EMI filters must be installed with the recommendations provided by the manufacturer, e.g., both filter and driver should be mounted securely on the same conducting earthed backplate. The supply cable should be a stranded conductor to achieve proper connection inside the terminal block. The cable between the driver and the motor should be a screened type and as short as possible with the screen and safety earth core connected directly to the bonded earth post at one end and to the motor earth at the other [10]. The conducting cores should be wound around an output cable filter choke. The paths through which earth current flows have to be clearly define. All earthing leads should be as short as possible and fastened to the bonded backboard earth post. Input and motor output cables should not run together in the same conduit and the control cables to the driver (or other signal cables) should not run alongside motor output cables.

C. Software Filtering

Sometimes not all the EMI can be eliminated by hardware, and software filtering is required to "erase" small particles of image noise [11]. Noise reduction techniques are very similar regardless of the signal being processed, although the characteristics of the expected signal make more suitable one technique or other. Both analogue and digital image signals are susceptible to noise. For example, in Fig. 2 "Video signal" is analogue and it is digitalized by the Matrox card. Noise can be random or white noise introduced by the devices mechanism or processing algorithms.

```
void CMDispMFCView::OnContOriginal()
{   theApp.m_pThread->m_pMainWnd->ShowWindow(SW_SHOW);
    PosFilter[0]=0; PosFilter[1]=0; PosFilter[2]=0;
 do{   OnGrabStop();
    MimBinarize(((CMDispMFCDoc*)GetDocument())->m_MilImage,
                ((CMDispMFCDoc*)GetDocument())->m_MilImage2,M_GREATER_OR_EQUAL,IMAGE_THRESHOLD_VALUE, M_NULL);
        else MimBinarize(((CMDispMFCDoc*)GetDocument())->m_MilImage, ((CMDispMFCDoc*)GetDocument())->m_MilImage2,
                M_LESS_OR_EQUAL,IMAGE_THRESHOLD_VALUE, M_NULL);
            MimOpen(((CMDispMFCDoc*)GetDocument())->m_MilImage,
                ((CMDispMFCDoc*)GetDocument())->m_MilImage2, 2*MIN_BLOB_RADIUS,M_BINARY);
    MimClose(((CMDispMFCDoc*)GetDocument())->m_MilImage2,
                ((CMDispMFCDoc*)GetDocument())->m_MilImage2, MIN_BLOB_RADIUS,M_BINARY);
            MimErode(((CMDispMFCDoc*)GetDocument())->m_MilImage2, ((CMDispMFCDoc*)GetDocument())->m_MilImage2, ErodeValue, M_BINARY);
            MIL_ID MilBlobFeatureList;
    MblobAllocFeatureList(((CMDispMFCDoc*)GetDocument())->m_MilSystem,&MilBlobFeatureList);
    MblobSelectFeature(MilBlobFeatureList, M_AREA);
    MblobSelectFeature(MilBlobFeatureList, M_CENTER_OF_GRAVITY);

    …
    MIL_ID MilBlobResult;
    MblobAllocResult(((CMDispMFCDoc*)GetDocument())->m_MilSystem,&MilBlobResult);
    MblobControl(MilBlobResult,M_IDENTIFIER_TYPE,M_BINARY);
    MblobCalculate(((CMDispMFCDoc*)GetDocument())->m_MilImage2,NULL,MilBlobFeatureList,MilBlobResult);
    MblobSelect(MilBlobResult,M_EXCLUDE,M_AREA,M_LESS_OR_EQUAL,MIN_BLOB_AREA,M_NULL);
    MblobGetNumber(MilBlobResult,&TotalBlobs);
  if ((CogX = (double *)malloc(TotalBlobs*sizeof(double))) && (CogY = (double *)malloc(TotalBlobs*sizeof(double))))
    { CogX[0]=0;CogY[0]=0;
    MblobGetResult(MilBlobResult, M_CENTER_OF_GRAVITY_X, CogX);
    MblobGetResult(MilBlobResult, M_CENTER_OF_GRAVITY_Y, CogY);}
 else    printf("not enough memory");
  if ((CorX = (double *)malloc(TotalBlobs*sizeof(double))) && (CorY = (double *)malloc(TotalBlobs*sizeof(double))))
    {CorX[0]=0;CorY[0]=0;
      if(FlowerDown==1)
        {MblobGetResult(MilBlobResult, M_X_MIN_AT_Y_MIN, CorX);
        MblobGetResult(MilBlobResult, M_BOX_Y_MIN, CorY);
        OffSetSign=-1;}
      else  {MblobGetResult(MilBlobResult, M_X_MAX_AT_Y_MAX, CorX);
        MblobGetResult(MilBlobResult, M_BOX_Y_MAX, CorY);
        OffSetSign=1;}}
 else    printf("not enough memory");
    From_pixel_to_dmm=ImageHeigth_dmm/XSize;
    long uni=1;
    min=0;
    for(n=1; n < TotalBlobs; n++)
    {if(CorX[n]<CorX[min]) min=n;   }
    uni=long((CorY[min]+OffSetSign*OffSet_dmm/From_pixel_to_dmm)*From_pixel_to_dmm-
        OffSet_Home_Image_dmm);
    if(uni<T_sup_min_dmm) uni=T_sup_min_dmm;            if(uni>T_inf_max_dmm) uni=T_inf_max_dmm;
    PosFilter[2]=PosFilter[1]; PosFilter[1]=PosFilter[0]; PosFilter[0]=uni;
    if (PosFilter[0]>=PosFilter[1] && PosFilter[0]<=PosFilter[2]) uni=PosFilter[0];
    if (PosFilter[0]<=PosFilter[1] && PosFilter[0]>=PosFilter[2]) uni=PosFilter[0];
    …
for(n=0; n < TotalBlobs; n++)
            {DrawCross(((CMDispMFCDoc*)GetDocument())->m_MilImage, CogX[n],
                CogY[n], CROSS_COLOR_WHITE*(Image_BN!=1)+CROSS_COLOR_BLACK*(Image_BN==1));
    CorY[n]=CorY[n]+OffSetSign*OffSet_dmm/From_pixel_to_dmm;
    if(CorY[n]<0) CorY[n]=0;
    DrawLine(((CMDispMFCDoc*)GetDocument())->m_MilImage, 0, CorY[n],
            XSize, CorY[n], CROSS_COLOR_WHITE*(Image_BN==1)+CROSS_COLOR_BLACK*(Image_BN!=1));
    DrawLine(((CMDispMFCDoc*)GetDocument())->m_MilImage, PixelSendCut, 0,PixelSendCut, YSize, CROSS_COLOR_WHITE*(Image_BN==1)+
CROSS_COLOR_BLACK*(Image_BN!=1));}
CROSS_COLOR_BLACK*(Image_BN!=1));}
    MblobFree(MilBlobResult);
            MblobFree(MilBlobFeatureList);
    int ij,ik,im,jhi;
    for(ij=0; ij< DelayCont; ij++)
    {for(ik=0; ik< DelayCont; ik++)
        {for(im=0; im< DelayCont; im++)
            {jhi=im;}}}
    OnGrabStart();
 } while(stopLoops);
```

Fig. 5 Core source code of the application program generated in Visual C++ based on MIL library. Important functions, constants and variables are in boldface

of the two thresholds, the pixel is set to the maximum value in the image; otherwise it is set to 0. The threshold values have to be selected to preserve the required information.

After binarization operation, the original noisy image acquired by the vision system (Fig. 4b) is transformed into a binarized image without noise (see Fig. 4c), after the application of image filtering presented in Fig. 5. This noise is typically due to the electromagnetic interference of the power amplifiers of electric actuators, e.g. AC or DC motors.

C. Erode and Dilate Process

The basic morphological operations [15], erosion and dilation, produce contrasting results when applied to either grayscale or binary images.Erosion shrinks image objects while dilation expands them. Erosion generally decreases the size of the objects and removes small parts by subtracting objects with a radius smaller than the structuring element. With grayscale images, erosion reduces the brightness (and therefore the size) of bright objects on a dark background. With binary images, erosion completely removes objects smaller than the structuring element and removes perimeter pixels from larger image objects.

Dilation generally increases the size of the objects, filling in holes and broken areas, and connecting areas that are separated by spaces smaller than the size of the structuring element. With grayscale images, dilation increases the brightness of objects by taking the neighborhood maximum when passing the structuring element over the image. With binary images, dilation connects areas that are separated by spaces smaller than the structuring element and adds pixels to the perimeter of each image object [16], [17].

Functions *MimErode()* and *MimDilate()* of Matrox Image Processing Library (MIL) can be used for basic erosion and dilation operations on 3 by 3 neighborhoods. The noise shown in Fig. 4b is removed after the binarization using these two functions. After this filtering, the Fig. 4c image is obtained.

D.Blob Analysis

The application developed in this work is based on this robust image analysis. Blobs are formed in the image when loops or a configuration of pen strokes get connected into an "ink blot" when the image is closed with a big structuring element [18]. This type of analysis is included in all image processing libraries, e.g., OpenCV, MIL, Sapera, etc. With this package, several important features for the detected blobs can be obtained: area; X and Y location of the centre of gravity; perimeter (normal and convex); size; Feret diameter (minimum, maximum, average) at specific angle; length; breadth; elongation (ratio of length to breadth); shape compactness (closeness of blob particles); number of holes in the blob; central and ordinary moments of any order; etc. For the application developed in this research, the features M_AREA and $M\_CENTER\_OF\_GRAVITY$ and the function *MblobSelectFeature()* of the MIL library are used. By means of this information, one or several elements (flowers/fruits/vegetables) can be located in the image and in

the real world with the camera calibration presented in Section IV.A. This blob analysis is used to reject the sheet shown in Fig. 4c and Fig. 4d images.
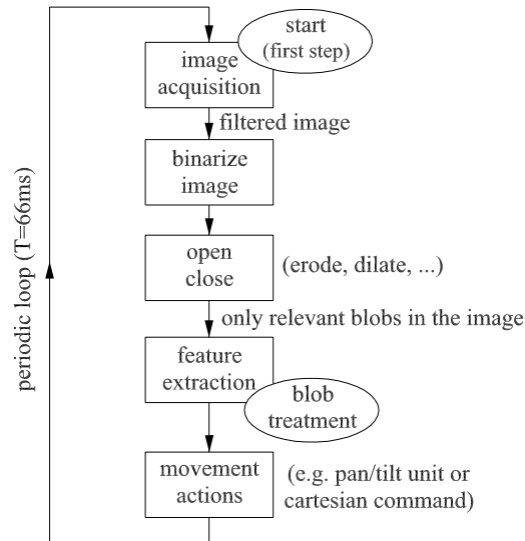

Fig. 6 Flowchart of the developed application

E. Computer Program Developed for Image Processing

Fig. 6 presents the flowchart of the developed application. The first step is image acquisition, which includes the software filtering described in Subsection "Erode and Dilate process". Next, the image is binarized. After that, the opening/closing operations and the erode/dilate algorithms are applied, hence only relevant blobs (i.e., the flowers/fruits/vegetables) remain in the image. The "feature extraction" step obtains the characterization of each detected blobs, i.e. the position of the centre of the flower/fruit/vegetable and the cutting point that allows to separate the element from its peduncle. Finally, the "movement action" step sends the characterization information of each flower/fruit/vegetable to the robotic mechanism responsible for the desired task: recollection, cutting, packaging, classification, fumigation, etc. The core source code of the developed program is shown in Fig. 5, which is included in the paper to facilitate the generation of new specific agricultural applications based on this image processing. This code has been generated in Visual C++ based on MIL library. The image could also be processed using Matlab [19], but in that case the computational cost is higher than using C++ and real-time many not be guaranteed.

The final application or program can be customized to work with different elements (flowers/fruits/vegetables), illumination conditions, element sizes/colours, etc. This flexibility is achieved through 17 configurable parameters, which are available in the "Configuration window". The most
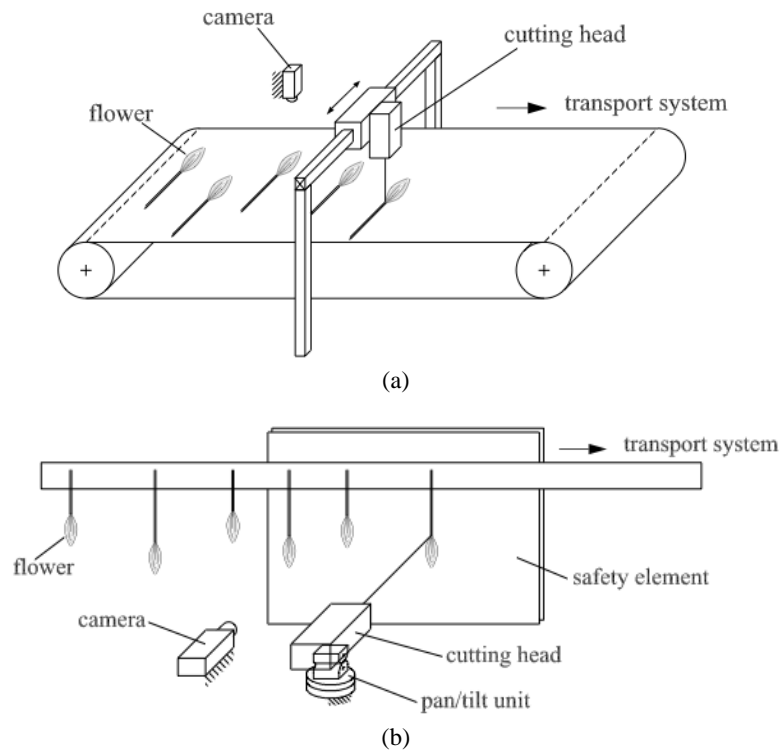
(a)

(b)

Fig. 8 Examples of applications for flower cutting with a (a) cartesian robot or a (b) pan/tilt robot: the image is processed and the cutting point
is obtained
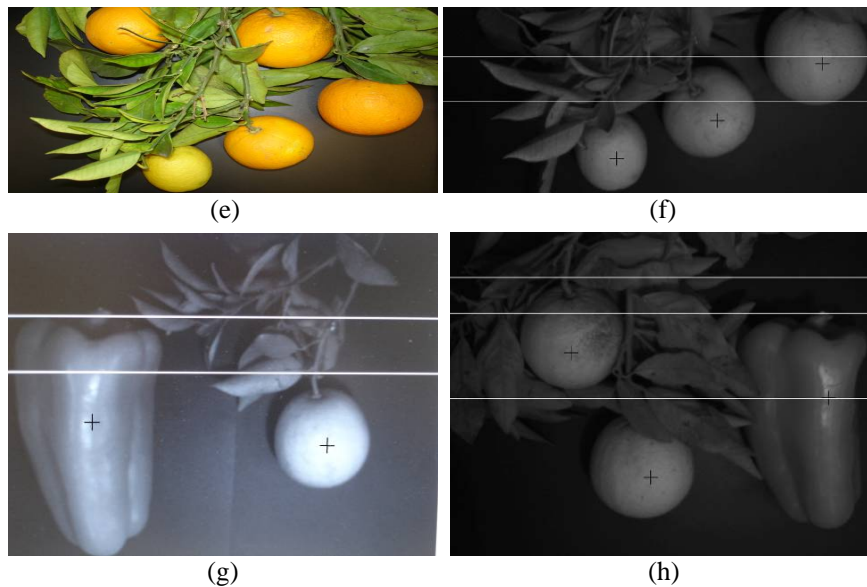

(e)

(f)

(g)

(h)

Fig. 7 Examples of the application performance. The program can be used with many types of flowers/fruits/vegetables, even simultaneously

important are:

1) *IMAGE_THRESHOLD_VALUE*: Threshold (from 0 to 255) used to binarize the acquired image. This value depends on the contrast, brightness, sharpness, etc. of the image.

2) *MIM_BLOB_AREA*: Minimum area to consider an object to be a blob.

3) *MAX_BLOB_AREA*: Maximum area to consider an object

to be a blob.

4) *Image_BN*: The value is set to 1 if the elements are darker than background (day); otherwise, the value is set to 0 (the elements are clearer than background, i.e. night).

5) *ErodeValue*: Number of pixels used to erode the detected object (flower/fruit/vegetable).

6) *OffSet_dmm*: Offset (tenths of millimetres) added/subtracted to the computed cut position.

V. RESULTS

The goal of the application is to compute the "centre of gravity" (position of the centre) of each detected object (flower, oranges, red pepper, etc.) and the cutting point that allows to separate the element from its peduncle. The computer program processes an acquired image with one element in about 50 milliseconds and processes an image with 8 elements in about 66 milliseconds. This can be considered real time from the processing machine (machine that physically manipulates/processes the elements) point of view.

Fig. 7a shows the appearance of the computer program and the experimental detection of a flower: the centre of gravity is depicted with a white cross and the peduncle cutting point is represented with a black horizontal line. Fig. 7b shows the processed image corresponding to the previous example. Fig. 7c and Fig. 7e show the original image acquired by the camera for other two examples with several oranges and sheets. As before, Fig. 7d and Fig. 7f depict the centre of gravity of each orange with a black cross and the cutting point for collecting them with a white horizontal line. Note that, in both examples the discrimination between oranges and sheets is done properly. The same happens in the examples of Fig. 7g and Fig. 7h (only the processed image is shown) where fruits and vegetables have been combined.

To illustrate the applications of the proposed program, two examples of flower cutting are shown in Fig. 8. In this sense, the program has been customized by the authors for saffron flower cutting giving rise to the commercial software SaffronCut® (2009) (protected by the Spanish office of software), which is currently being used by a Spanish saffron company. In fact, the program described here is intended to be the base for developing new applications that use computer vision in specific agricultural tasks (such as recollection, cutting, packaging, classification, fumigation, etc.) both in the country and in warehouses. That is why this paper emphasizes and solves some important practical problems, such as the EMI, and provides the core source code of the application.

VI. CONCLUSION

In this research, a new vision system to characterize the position of flowers/fruits /vegetables in complex images has been developed. This information can be used for several types of agricultural tasks, such as recollection, cutting, packaging, classification, fumigation, etc. The tests made with the prototype in the laboratory evidence the feasibility and validity of the proposed approach. In particular, the results

show that it has good robustness and a very high success. An important characteristic of the proposed algorithm is that it is able to work with several elements inside the camera field of view. This adds flexibility to the application in order to work in the country or in a greenhouse, where the elements are very close to each other and the location of all them must be obtained in real time. We conclude that the use of machine vision in agricultural applications could be increased using the proposed approach, improving the automation of the production and the profits.

REFERENCES

[1] FruitLogistica informs, 2003-2008. Agrooglyad: Vegetables and Fruits. http://www.lol.org.ua/eng/

[2] P. C. Condlife, M. R. Davey, B. J. Power, H. Koehorst-Van Putten, and P. B. Visser, "An optimised protocol for rose transformation applicable to different cultivars," in *XXI International Eucarpia Symposium on Classical versus Molecular Breeding of Ornamentals*, München, 2003.

[3] J. Blasco, N. Aleixos, J. Gomez, and E. Molto, "Citrus sorting by identification of the most common defects using multispectral computer vision," Journal of Food Engineering, vol. 83, no. 3, pp. 384–393, 2007.

[4] J. Blasco, S. Cubero, J. Gomez-Sanchis, P. Mira, and E. Molto, "Development of a machine for the automatic sorting of pomegranate (Punica granatum) arils based on computer vision," Journal of Food Engineering, vol. 90, no. 1, pp. 27–34, 2009.

[5] K. S. Fu, R. C. Gonzalez, and C. S. G. Lee, *Robotics Control, Sensing, Vision and Intelligence*. New York: McGraw-Hill, 1987.

[6] W. Niblack,. *An Introduction to Digital Image Processing*. New Jersey: Prentice Hall, 1986.

[7] A. G. Manh, G. Rabatel, L. Assemat, and M. J. Aladon, "Automation and emerging technologies: weed leaf image segmentation by deformable templates," Journal of Agricultural Engineering Research, vol. 80, no. 2, pp. 139–146, 2001.

[8] D. Bulanon, T. Kataoka, H. Okamoto, and S. Hata, "A Real-time Image processing algorithm for apple fruit detection," Journal of Hokkaido Branch of the Japanese Society of Agricultural Machinery, vol. 45, pp. 71–75, 2005.

[9] J. Blasco, N. Aleixos, J. M. Roger, G. Rabatel, and E. Molto, "Robotic weed control using machine vision," Biosystems Engineering, vol. 83, no. 2, pp. 149–157, 2002.

[10] A. N. Hejase, A. T. Adams, R. F. Harrington, T. K. Sarkar, "Shielding effectiveness of `pigtail' connections," IEEE Transactions on Electromagnetic Compatibility, vol. 31, no. 1, 63–68, 1989.

[11] E. R. Dougherty, *An Introduction to Morphological Image Processing*. Washington: SPIE Optical Engineering Press, 1992.

[12] H. C. Raymond, H. Chung-Wa, and N. Mila, "Salt-and-pepper noise removal by median-type noise detectors and detail-preserving regularization," IEEE Transactions on Image Processing, vol. 14, no. 10, pp. 1479–1485, 2005.

[13] K. M. Schmitt, R. C. D. Young, J. R. Riddington, D. M. Budgett, C. R. Chatwin, "Image processing applied to brick quality control," The International Journal of Advanced Manufacturing Technology, vol. 16, no. 6, pp. 434–440, 2000.

[14] Y. Abdel-Aziz and H. Karara, "Direct linear trasnformation from comparator coordinates into object space coordinates in close-range photogrammetry," in *Symposium on Close-Range Photogrammetry*, Illinois, 1971, pp. 1–18.

[15] J. Serra, *Image Analysis Using Mathematical Morphology*. New York: Academic Press Inc., 1982.

[16] R. M. Haralick, S. R. Sternberg, and X. Zhuang, "Image analysis using mathematical morphology," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 9, no. 4, pp. 532–550, 1987.

[17] S. Nagabhushana, *Computer Vision and Image Processing*. New Age International. 2006.

[18] T. Kanungo and R. M. Haralick, "Character recognition using mathematical morphology" in *Proceedings of the 5th USPS Advanced Technology Conference*, Washington, 1990, pp. 237–251.

[19] R. C. Gonzalez, R. E. Woods, and S. L. Eddins, *Digital Image Processing Using MATLAB*. New Jersey: Prentice Hall, 2004.