ELSEVIER Computers and Electronics in Agriculture 13 (1995) 301–314

Computers and electronics in agriculture

A thinning-based algorithm to characterize fruit stems from profile images

F. Pla [a,*], F. Juste [b]

[a] *Departament d'Informàtica, Universitat Jaume I, 12071 Castellón, Spain*
[b] *Instituto Valenciano de Investigaciones Agrarias, Apartado Oficial, 46113 Moncada Valencia, Spain*

Accepted 15 May 1995

Abstract

In this paper we present a thinning-based approach to detect protrusions from binary patterns, and its use to characterize fruit stems from fruit profile images. The method to detect protrusions in binary patterns is based on the property of iterative thinning algorithms which causes protrusions to become skeletons during the first iterations of the thinning process. A modified thinning method, and a restoration of the thinned parts using criteria based on some topological properties of digital images, allow to identify, locate and characterize in size and length these protrusions from binary patterns.

The method has been applied to characterize fruit stems, allowing the successful detection of a wide range of stem shapes, with high effectiveness, 99% of stems being correctly identified.

Keywords: Machine vision; Thinning; Stem location

1. Introduction

The presence of stems in fruits may be either a problem or desirable, depending on the commercial purpose and the type of fruit. Stems produce fruit damage during handling and stocking processes, because of puncture injuries, for example in tomatoes and oranges. Nevertheless, stems are desirable for some fresh fruits to avoid drying or decay during transport to market.

Sometimes stem is desirable and increases product quality when used for certain purposes and sometimes, for the same fruit, the presence of stem is considered as a negative quality factor, for example in cherry peppers. These two opposite situations are also present when the use of fruits has to be optimized, depending on whether

* Corresponding author.

SSDI 0168-1699(95)00029-1

they have stems or not, for example, when fruits like blueberries with stem are used for fresh market, and those without stem are used for food industries (pastry, ice cream, etc.).

Mechanical systems have been developed to de-stem fruits like oranges (Chen, 1994), strawberries (Kirk and Booster, 1978), tomatoes (Giacomelli and Studer, 1981) or onions (LePori and Hobgood, 1970). The techniques used in these mechanical systems are cutting, counter rotating rollers and impact. In general, these mechanical systems produce a certain degree of bruise damage, and they do not achieve complete de-stemming of all fruits.

Another approach is the use of computer vision to recognize and locate the stems automatically, and thus perform an accurate de-stemming. Following this principle, Wolfe and Sandler (1985) developed an algorithm to detect stems using binary profile images of blueberries and cherry peppers. This method was based on the analysis of contours from fruit profile images, using a syntactic pattern recognition technique. From another point of view, Yang (1993) used structured light techniques to detect stalk-calyx areas in apples, from images of randomly oriented fruits.

The motivation of the work we are presenting here has been the detection and characterization of orange stems to be eliminated before fruits are stocked in boxes. In automatic harvesting of citrus fruits by a picking robot, around 35% of harvested fruits, depending on variety, have stems (Fornes et al., 1994). After fruits have been picked they are stored in fruit containers; therefore, a certain proportion, 7% approximately, of fruits suffer damage by puncture injuries. In order to avoid this damage, fruits have to be de-stemmed before they are stored.

Fruits picked by a robotic arm have quite often stems with several shapes (T-shapes, L-shapes, etc.) and size, and they occasionally have some leaves attached to the stem. The approaches made so far to detect stems in fruits cannot deal with these situations, thus, the aim is to develop an image analysis technique capable of handling the problem described. A machine vision technique is proposed here to detect, measure and locate stems in oranges, as part of a de-stemming system for fruits picked by a robotic arm. The method assumes that fruit profile images can be obtained, showing the stem as a protrusion from the fruit body.

The proposed method consists of a modified thinning technique, which makes is possible to identify, locate and characterize any fruit protrusion. In contrast to some filtering techniques applied to binary patterns, like mathematical morphology filtering, the proposed method does not erase protrusions in the binary patterns. It maintains the body and the contour of the original pattern unchanged, while transforming the protrusions into skeletons. From the skeletons of the protrusions, the method can locate the point where stems join the fruit, and estimate stem shapes and sizes.

The rest of the paper is organized as follows. After the principles of the method are outlined, the algorithm designed to detect the stems is explained. Once stems are identified, the subsequent section describes how the point where the stem joins the fruit is located, and the length and size of the stem are estimated. Finally, after the presentation of some experimental results, the conclusions drawn from this work are described.

2. Algorithm principles

Thinning algorithms in image analysis are used to extract the skeleton of objects from binary images. The term skeleton has been used in image analysis to denote a representation of a pattern by a collection of thin arcs and curves (Fig. 1). Iterative thinning algorithms delete successive layers of pixels from the boundary of the patterns until only a skeleton remains. In this case the pattern is a region formed by connected pixels in a binary image.

Given a binary pattern from a profile image of a fruit showing a stem, the human eye can easily distinguish between the part of the pattern which represents the fruit body, as a compact region, and a protrusion appended to the fruit body, formed by an elongated pattern much thinner than the fruit body. In general, any protrusion from the fruit body, like leaves attached to the stem, will have a thinner pattern than the fruit body.

Therefore, if we analyze the behavior of an iterative thinning algorithm applied to a profile image of a fruit with stem, we realize that the thinner parts of the pattern, that is, the stems, become skeleton during the first iterations of the thinning. Thus, it would be possible to characterize, and then to identify, any protrusion from the fruit body analyzing what happens during the first iterations of a thinning, in such a way that it allows us to identify, locate and estimate the length and shape of the stem, and to decide whether a fruit has stem or not.

3. Detecting fruit protrusions

Following the principles described in the previous section, the first step to achieve the identification of fruit stems is to perform an iterative thinning algorithm over the binary pattern of a fruit profile image, obtained from thresholding of a gray level image of a fruit. A survey of the standard thinning algorithms in the literature can be found in Lam et al. (1992).

As we have already pointed out, thinner parts of the pattern become skeleton quickly during the first iterations of the algorithm, thus the thinning process is stopped after N iterations, with N being a number depending on the expected thickness of stems in the images. Therefore, after these iterations stems become

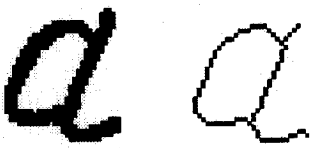

Fig. 1. A binary pattern of the 'a' letter, and its skeleton found by thinning.

x8	x1	x2
x7	P	x3
x6	x5	x4

Fig. 2. $\times 1, \ldots, \times 8$ are the 8-connected neighbors of p. $\times 1$, $\times 3$, $\times 5$ and $\times 7$ are the 4-connected neighbors of p.

skeletons, but avoiding that the fruit body becomes skeleton. As we will explain later, it is important that the fruit body does not reach the skeleton state, because the recognition of the stem is based on the different characterization between the parts which have become skeleton during the first N iterations, and those which have not reached the skeleton state. To differentiate pixels from a pattern to know whether they are skeleton pixels or not, we have to evaluate some properties of each pixel.

The decision the thinning algorithms make to delete a boundary pixel at each iteration, is based on the topological properties of each pixel. Topological properties are defined from the relation between a given pixel and its neighbors. We are considering patterns whose pixels are 8-connected, that is, each pixel in the image grid is considered to have 8 neighbors (Fig. 2), thus, two pixels are 8-connected if they are 8-neighbors. Pixels $\times 1$, $\times 3$, $\times 5$ and $\times 7$ in Fig. 2 are also called the 4-neighbors of pixel p.

Therefore, a pixel p belonging to the pattern, is said to be 8-connected to pixel q, which also belongs to the pattern, if there is a path from p to q consisting of pixels belonging to the pattern. A path from pixel c_1 to the pixel c_n is a sequence of pixels $c_1, c_2, \ldots c_{n-1}, c_n$, such that the pixel at c_k is 8-connected to the pixel c_{k+1}, for all k with $1 \leq k \leq n$. The same applies for pixels 4-connected when we assume 4-connection instead of 8-connection.

According to these topological properties, pixels belonging to the pattern can be classified into the following classes:

- *body pixels:* pixels from the pattern that, if we deleted it, its neighbors would remain 8-connected. These body pixels can be divided into two classes:
  - *boundary pixels:* body pixels which have some 4-neighbor which do not belong to the pattern.
  - *interior pixels:* body pixels which have all their 4-neighbors belonging to the pattern.
- *skeleton pixels:* pixels from the pattern that, if we deleted it, its neighbors would not be 8-connected. These skeleton pixels can be divided into four classes:
  - *end pixels:* pixels which have a single 8-neighbor belonging to the pattern.
  - *arc pixels:* skeleton pixels which are 8-connected to two pattern pixels.
  - *T-pixels:* skeleton pixels which are 8-connected to three pattern pixels.
  - *X-pixels:* skeleton pixels which are 8-connected to four pattern pixels.

Therefore, according to these definitions, and to differentiate between skeleton and body pixels, during the first N iterations of the thinning, the pixels the thinning

algorithm decides to maintain, are labeled either as skeleton or body pixels. After N iterations we could identify stems, or protrusions, as the parts of the pattern labeled as skeleton pixels.

Unfortunately, this assumption would lead to many mistakes, since during the thinning iterations, any sharp local configuration in the boundary of the pattern leads to a branch of the final skeleton of the pattern (Figs. 3 and 4).

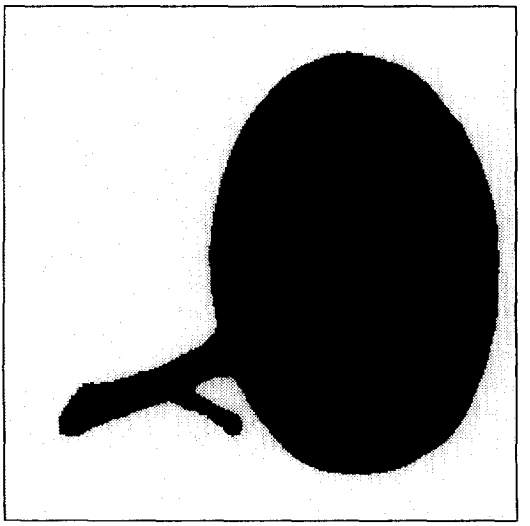

Fig. 3. Original binary pattern of an orange profile image.

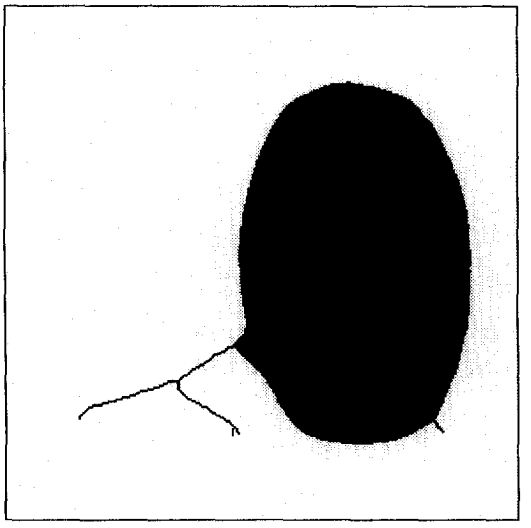

Fig. 4. Binary pattern from Fig. 3 after stopping the thinning at 10 iterations.

3.1. Undoing the thinning

To avoid the above mentioned problem, after the first N iterations of the thinning algorithm, we could undo the thinning performed by adding layers of pixels in such a way that the fruit body could be completely recovered, but leaving any protrusion, like the stems, as skeletons, allowing then an easy identification of the stems.

In order to recover the same layers of pixels deleted during the N iterations of the thinning, at each iteration during the thinning, a list of the pixels deleted from the pattern is saved. Therefore, at the end of the N iterations, we have stored N list of pixels, ordered according to the number of the performed iterations.

If we undo the thinning restoring the layers of pixels saved, starting from the last thinning iteration, we would recover the original pattern. Since this is not the pursued goal, a criterion is established each time we consider a pixel to be restored to decide whether it is restored or definitely discarded. This criterion has to be established in such a way that at the end of the restoring process the fruit body is completely restored and any protrusion like stems remains skeleton.

The criterion to decide whether a pixel is restored or not is based on the idea that only layers of pixels can be added to the boundary of the pattern formed by body pixels, that is, only body pixels can generate body pixels. Thus, considering the list of pixels saved from the last iteration in the thinning step, the pixels restored to the pattern are labeled as body pixels.

It seems obvious that if only pixels can be added to the boundary of the pattern formed by body pixels, the criterion established to restore a pixel being considered is that only pixels which have an 8-connected neighbor labeled as body pixel can be restored.

After restoring the N lists of pixels, in reverse order with respect to the order they were saved during the thinning process, and taking into account the described criterion to decide whether a pixel is restored or not, parts of the original pattern corresponding to protrusions from the fruit body remain skeleton, and thus they can be identified and located (Fig. 5), since at the end of the process, pixels of the pattern are labeled either as body or as skeleton pixels.

3.2. The algorithm

Let us summarize in the following algorithm the process described so far to detect and identify fruit stems or any other protrusion emerging from the fruit body:

```
Initialization: Label all pixels in the pattern as body pixels;
/* First stage: performing the thinning up to N iterations */
For i := 1 to N
    For each pixel in the pattern
        check label of pixel neighbors;
        If the pixel can be removed
            append the pixel to list i;
            remove the pixel from the pattern;
```

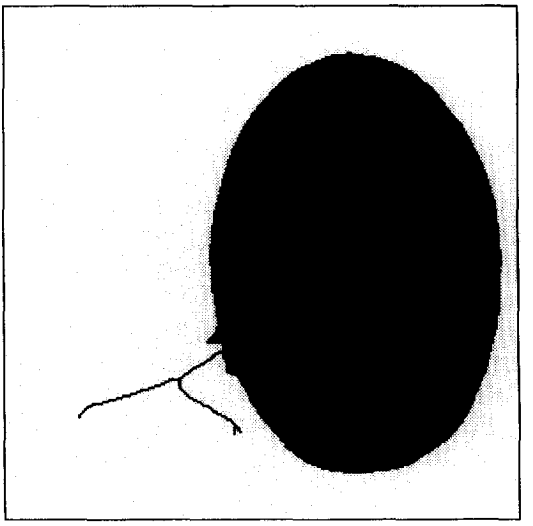

Fig. 5. Binary pattern from Fig. 3, after restoring the thinning by the method proposed. Skeleton pixels are represented clearer than body pixels. The joining point found between the stem and the fruit body is also shown.

```
        else
            check the connectivity properties of the pixel;
            label the pixel either as skeleton or body pixel according to its proper-
            ties;

/* Second stage: undoing the thinning */
For i := N to 1
    For each pixel in list i
        If it has any 8-neighbor labeled as body pixel
            restore the pixel and label it as body pixel;
End.
```

4. Locating the stem

Once the stem, or any protrusion from the fruit body, has been characterized and detected, it has to be located in order to supply this information to the de-stemming mechanism, in order to take the stem off in an accurate and selective way.

The most representative point of reference to locate the stem for the de-stemming system, is the point where the stem joins the fruit. Recall that after the stem detection algorithm, the pattern of the fruits is formed by the fruit body, whose pixels are labeled as body pixels, and the skeletons of the fruit protrusions, labeled as skeleton pixels. Each protrusion skeleton is united to the body from a pixel. Therefore, this skeleton pixel has one or more body pixels in its neighborhood.

Thus, to locate the point where the stem joins the fruit, we have to check

each skeleton pixel of the pattern, and identify the one which has body pixels in its neighborhood. The coordinates of this point are supplied to the de-stemming system (Fig. 5).

5. Measuring the size of the stem

Apart from the problem of detecting whether a fruit has a stem or not, and what is the point where the stem joins the fruit, it is a valuable information how large is the stem, in length and size, in order to finally decide if it has to be eliminated, since very short stems do not need to be suppressed.

A method to estimate the length of the stem is to measure the length of the skeleton in the pattern corresponding to the stem. The length of the skeletons can be considered, with little error, proportional to the real length of the stem. Therefore, stem lengths can be compared with their skeleton lengths. The length of a skeleton is defined as the number of connected pixels belonging to this skeleton. They can be counted from the image pattern resulting after applying the stem detection algorithm.

Another feature to characterize the stem is its size, considering the size of the stem in the image as the area of the part in the original pattern corresponding to the stem. This area can be measured if we segment the original binary pattern into two regions; the fruit and the stem. To perform this segmentation, remember that the result of the stem detection algorithm is a pattern in which we can distinguish between the body and skeleton pixels. Since the body pixels represent to the fruit body completely restored, we can perform the following logical operation between the original pattern and the pattern resulting from the algorithm previously explained:

For each pixel in the original pattern *If* it is labeled as body pixel in the resulting pattern label it as fruit pixel; *else* label it as stem pixel;

The result of this operation segments the original pattern into the two types of regions searched (Fig. 6). The area of the region corresponding to the stem provides an estimation of the stem size.

6. Experimental results

A set of 124 gray level images from orange profile views were used to test the algorithm. The images were taken with a CCD monochrome camera and digitized in a frame grabber installed on a PC bus. The digitizer board acquired images of $512 \times 512 \times 8$ bits pixel resolution, although images were reduced to 256×256. The images were taken using an illumination chamber, providing diffuse illumination to the fruits to avoid shadows and specular reflections. To obtain the binary patterns images were thresholded.

From the 124 images 29 did not have stem and the rest, 95, showed stems of different size and shape. Some of the fruit images with stem showed leaves attached to the stem, in order to consider such a case, because they occasionally occur during the automatic harvesting of fruits by a robotic arm (Fornes et al., 1994).

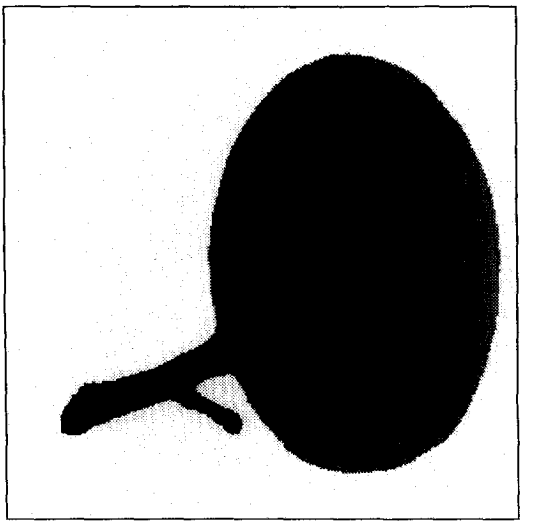

Fig. 6. Result of the segmentation to differentiate between the fruit and stem area of the binary pattern from Fig. 3.

Table 1
Result of the algorithm on the test image set

Fruit with stem		Fruit without stem	
Correct	Errors	Correct	Errors
93	2	29	0

The result of applying the stem detection algorithm to the 124 images is shown in Table 1. The algorithm detected correctly all the stems from the images of fruits with stem, and it did not detect any stem in the 29 images which did not have stems. Two images of fruits showing stems are presented in Table 1 as mistakes. In these two images the algorithm detected the stem and another protrusion, in this case they detected the blossom end of the fruit which was slightly sharp and the algorithm detected it. Although these two cases were considered an error, it would be possible to establish some criterion to choose one of the protrusions detected by the algorithm to assign it to the stem. This criterion could be based on the stem length, since blossom end protrusions are usually very short, and they are not as sharp as the stem protrusions are. Fig. 7 shows one of these two cases where the stem was detected, and the blossom end of the fruit was also detected as a protrusion.

In Fig. 8, an example of leaves attached to the stem is shown. Note how the algorithm has tackled the problem, being successful in such a case too. In this type of scenes problems could arise to determine the point where the stem joins the fruit, since sometimes the profile image shows the leaf overlapped to the fruit, so, in these cases the skeleton of the stem and leaves may have more than one point of contact

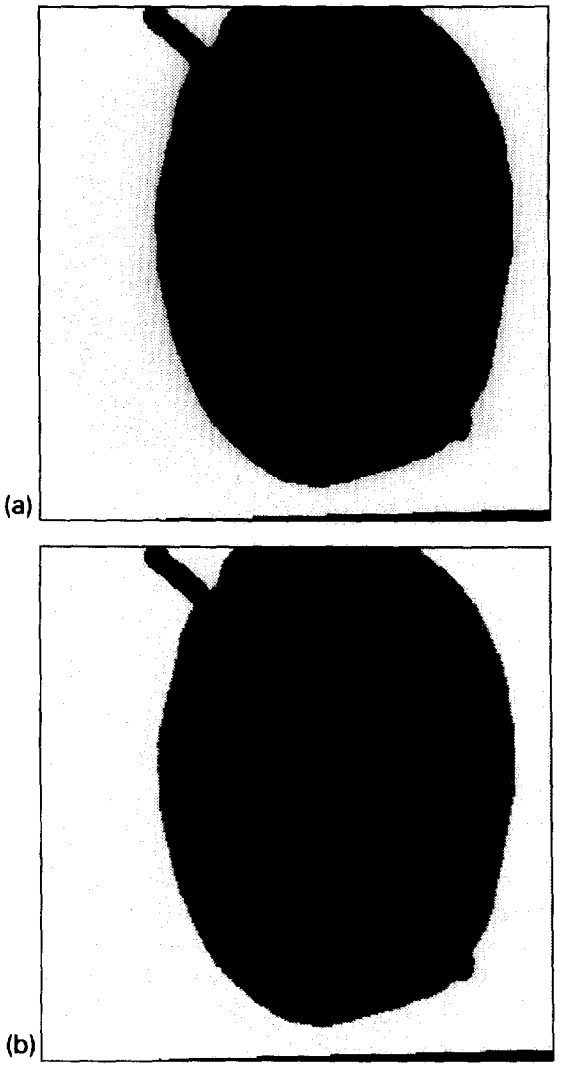

Fig. 7. Result of the stem detection algorithm on a fruit showing a sharp blossom end. (a) Original binary pattern; (b) result.

with the fruit body. In any case, the algorithm can deal correctly with most of these cases.

Fig. 9 shows the result of the algorithm on an image of a fruit showing a short stem, and Fig. 10 represents the result on an image of a fruit which did not have stem. Comparing the examples shown with larger stems (Fig. 5) and shorter stems (Fig. 9) we can conclude that estimating the length of the detected stems by their skeleton length is a coherent approximation.

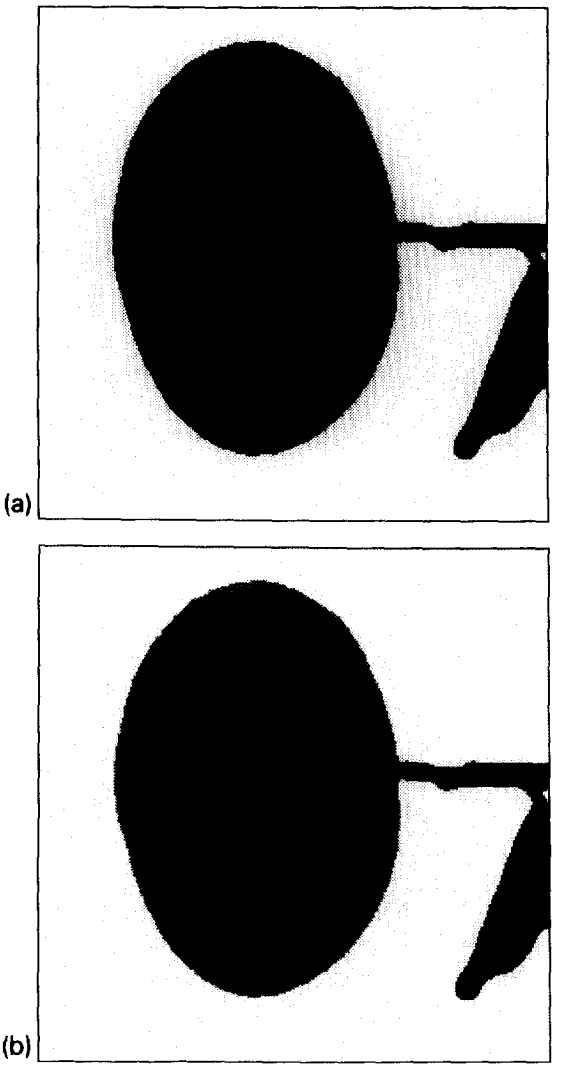

Fig. 8. Result of the stem detection algorithm on a fruit showing leaves attached to the stem. (a) Original binary pattern; (b) result.

Overall, the most important feature of the algorithm is that it can detect stems from profile images with any shape and size, locating them and giving an estimation of their size. With respect to the computational time, although the algorithm in its present implementation takes a short time, it could be implemented in a parallel machine to achieve real-time performance, since the most expensive part of the algorithm is the first stage, corresponding to the thinning, and there are parallel thinning algorithms which can perform at video rate (Jang and Chin, 1992).

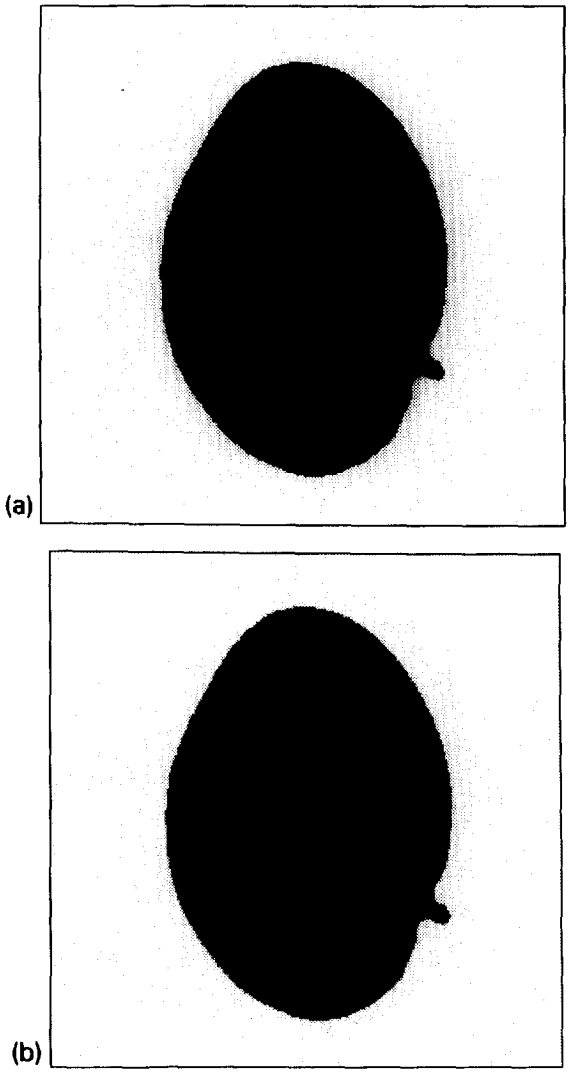

Fig. 9. Result of the stem detection algorithm on a fruit showing a short stem. (a) Original binary pattern; (b) result.

7. Conclusions

An algorithm based on thinning techniques has been presented to detect, locate and estimate the length of fruit stems from binary profile images. The algorithm uses the shape information provided by the pattern skeletons to characterize and identify the stems or any protrusion from the fruit body.

The algorithm can deal with profile images showing stems of different shape

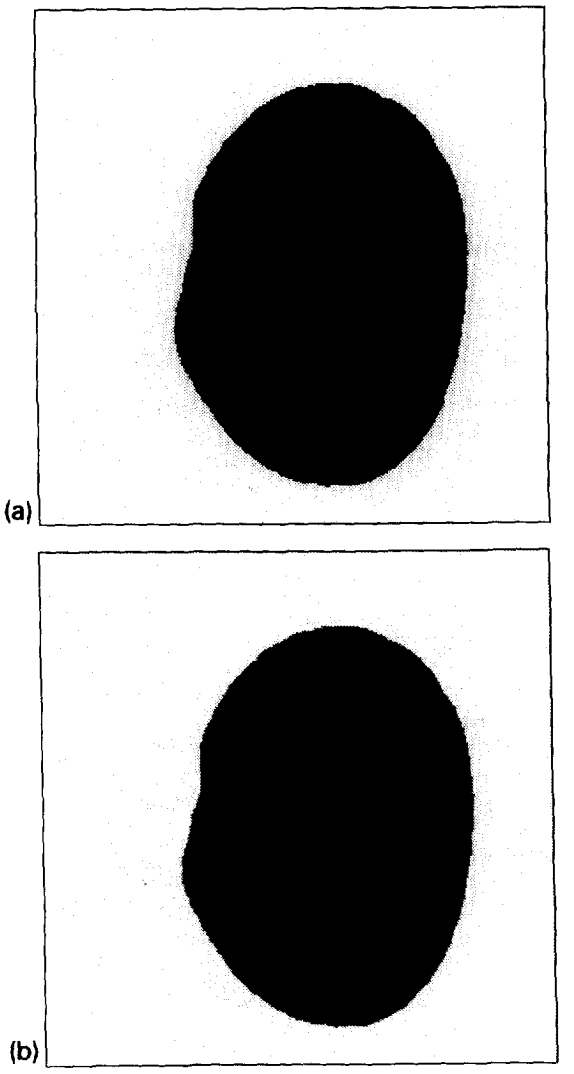

Fig. 10. Result of the stem detection algorithm on a fruit without stem. (a) Original binary pattern; (b) result.

and size, and it even detects stems which still have leaves attached. Although the presence of leaves does not affect the stem detection algorithm, it can occasionally mislead the location of the point where the stem joins the fruit, because of the shape of the binary pattern.

The algorithm can detect stems with negligible errors. Although it assumes to have profile images of fruits, these images can be obtained using a fruit handling device controlled to get a map of the profile views of a fruit.

Real-time performance could be achieved implementing the thinning algorithm in a parallel machine, or a hardware specific image processing a module. The rest of the algorithm can be performed sequentially by a host computer without significant computational cost.

Acknowledgments

This work has been partially supported by EUREKA project EU-176, and INIA project 9513.

References

Chen, P. (1994) Mechanical destemming of oranges. In: F. Juste (Editor), Fruit Nut and Vegetable Engineering Production, Vol. 2, pp. 205–210.

Fornes, I., Juste, F., Santamarina, C. and Bimbo, B. (1994) Potential damage to citrus fruits in mechanical harvesting. In: F. Juste (Editor), Fruit Nut and Vegetable Engineering Production, Vol. 2, pp. 51–60.

Giacomelli, G.A. and Studer, H.E. (1981) Orienting and stemming mature green tomatoes. Trans. ASAE, 24(4): 884–888.

Jang, B.K. and Chin, R.T. (1992) One-pass parallel thinning: analysis, properties and quantitative evaluation. IEEE Trans. Pattern Recognition Mach. Intell., 14(11): 1129–1140.

Kirk, D.E. and Booster, D.E. (1978) Capping and stemming mechanically harvested strawberries. ASAE Paper 78-6537.

Lam, L., Lee, S. and Suen, C.Y. (1992) Thinning methodologies: a comprehensive survey. IEEE Trans. Pattern Recognition Mach. Intell., 14(9): 869–885.

LePori, W. and Hobgood, P. (1970) Mechanical harvester for fresh market onions. Trans. ASAE, 13(5): 517–519.

Wolfe, R.R. and Sandler, W.E. (1985) An algorithm for stem detection using digital image analysis. Trans. ASAE, 28: 641–644.

Yang, Q. (1993) Finding stalk and calyx of apples using structured lighting. Comput. Electron. Agric., 8(1): 31–42.