ELSEVIER

Image and Vision Computing 14 (1996) 353–363

image AND vision COMPUTING

Structure from motion techniques applied to crop field mapping

J.M. Sanchiz[a], F. Pla[a], J.A. Marchant[b], R. Brivot[b]

[a]*Department of Computer Science, University Jaume I, 12071 Castellon, Spain*
[b]*Silsoe Research Institute, Wrest Park, Silsoe, Beds MK45 4HS, UK*

Received 10 July 1995; revised 24 October 1995

Abstract

Some agricultural tasks performed in a crop field consist of applying chemical treatment to the plants. To automate these tasks, a vision system and a treating device can be used as a sensing device, mounted at the rear of an agricultural vehicle. The vision system tracks each plant in a sequence of images. With this arrangement the position of each plant with respect to the treating device can be derived, which specifies when and where to apply the chemical spray. A map of the field has to be recovered in the local area between the vision sensor and the treatment device. This implies recovering the motion parameters of the vehicle, recording the trajectory of the vehicle, and placing the plants in the map. A method to identify the plants, a shape description to match them and a tracking strategy are presented. The motion parameters are recovered from the plant correspondences, using a method to find the motion of a planar patch. A Kalman filter is used to integrate the different observations of each plant and to place them on the map. Results with real image sequences are presented, including zigzag and rotational movement.

Keywords: Motion analysis; Tracking; Kalman filter

1. Introduction

Some agricultural tasks performed in crop fields consist of applying chemical treatment to the plants. To automate these tasks, a vision system and a treating device could be mounted on the rear of an agricultural vehicle. While the vehicle moves, the vision system has to identify and track the plants, in order to apply the treatment accurately. The system could be mounted as a trolley towed by a manually driven vehicle, or it could be a part of an autonomous agricultural vehicle. In this case, the motion parameters recovered by the system would also be useful information for the guidance control system. The exact positions of the plants with respect to the vehicle would have to be known, which implies building a map of the field from the different observations of the plants with respect to the position of the vehicle. Information received from the tracking system would be used by a task planning module that would decide which plant has to be treated and when.

Vehicle navigation has one of its practical applications in agricultural task automation. Vision-based guidance methods for vehicle navigation have been reported in recent years in the problems of road following [1–4] and indoor navigation [5,6]. The problem of analysing camera motion from a sequence of images has also been studied widely. In vehicle navigation applications or robot motion analysis, the scene is supposed to be static while the camera is moving. Feature-based methods use collections of matched features in pairs of consecutive images [7–12]. Features are usually points or lines, and the matching between features is made using some similarity measurement, for example, correlation of a small window around corners. Camera motion analysis is closely related to the problem of the tracking of tokens, and structure from motion. The Kalman filter and the extended Kalman filter have been used to integrate the different observations of the features, and to estimate their positions or depths [13–16].

In this application there are no man-made objects to focus on. Typical scenes we are dealing with consist of a piece of field with some plants, weeds and soil. First, a segmentation process is applied to separate these three classes. To get a good contrast between vegetation and soil, grey level images of the field are taken with an infrared filter, then the images are segmented with a grey level threshold [17]. Two-dimensional regions obtained from the segmentation process are the input data to the vision system. These regions form the plants.

Since our main interest is to solve the practical

SSDI 0262-8856(95)01070-X

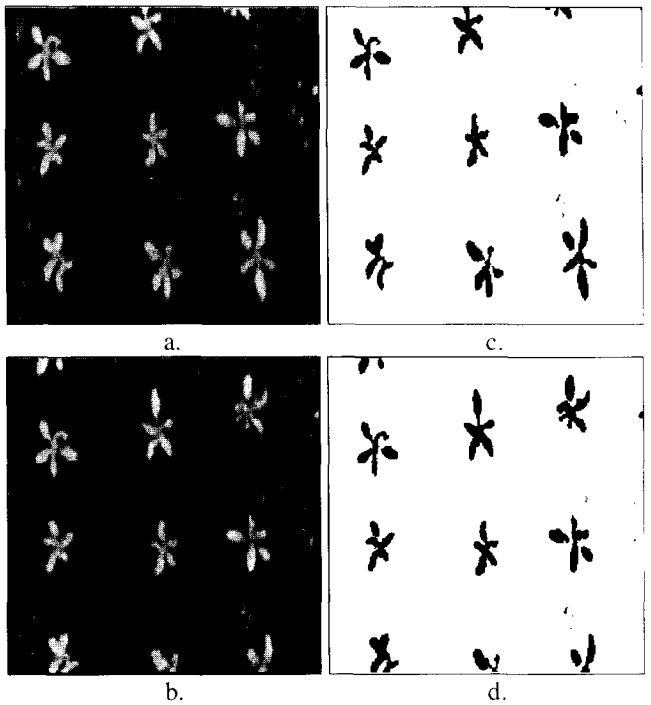

Fig. 1. Grey level and segmented images. (a) First image of a sequence, (b) next image, (c) and (d) segmented images.

problem of tracking plants in a crop field, and the objective is to have a real-time implementation, we have to use non computationally expensive algorithms. In contrast to other approaches related to token tracking, in which the features to track are corners or line segments [7–19], we decided to use a high level approach, and to track whole objects like plants. A simple shape description for plants will help us to perform the tracking. The motion between subsequent images can be large, so solutions based on difference images have to be discarded. Each object appearing in the image has to be identified, and matched with one object of the previous image. These matches will be used to recover the motion parameters of the vehicle, and the map of the field. The centres of the plants will be used as a set of corresponding points in two images.

To estimate the exact positions of the plants in the field, a Kalman filter is applied to each one as it appears in subsequent images. The Kalman filter has been widely applied in computer vision to the problem of estimating unknown positions or movement over time from a set of observations in subsequent images. Reported work includes estimation of the depth of feature points in a static scene while the camera moves [18,19], and estimation of motion parameters in token tracking while the camera remains static [20,21]. In our case the scene is considered to be static and the movement is restricted by the geometry of the problem: images are taken from a vehicle moving over the ground, with the image plane nearly parallel to the ground plane. The motion parameters of the camera are *a priori* unknown. They are estimated from matches between objects that appear in subsequent images through a least-squares minimisation method. These motion parameters are used in the Kalman filter. As a result of the restrictions applied to the movement, the Kalman filter implementation is very simple, and well suited for real-time purposes.

The steps performed to obtain a map of the field are, for each pair of subsequent images:

- Segment the infrared image.
- Identify each plant.
- Compute a shape description for each plant and track the plants.
- Compute the motion parameters of the vehicle.
- Update the vehicle position in the field.
- Place the plants in the field by integrating their observations.

Grey level and segmented images can be seen in Fig. 1. Fig. 1(a) shows the first image of a sequence, and Fig. 1(b) the following one. Figs. 1(c) and (d) show the segmented images.

The rest of this paper is organised as follows. First, a method for the identification of plants is presented. Second, a shape description for them and a tracking strategy are presented. The third section describes a method to recover the motion of the vehicle. The fourth section deals with recovering the positions in which plants are placed in the field, and so the field map. Finally, results with real image sequences are presented.

2. Identification of plants

The difficulty of identifying and tracking plants relies on the fact that plants are three-dimensional objects, viewed from different perspectives as the vehicle is moving. Segmentation errors may occur, and the number and shape of regions that form each plant in the segmented image can change from one image to another.

After the first processing stage, consisting of segmenting the image, each plant appears as a collection of a small number of 2D regions. These regions can split or merge from one image to another, although the regions that form each plant are grouped close to the centre of the plant; so, plants can be identified by clustering the regions that appears in the scene, and then each cluster of regions is identified as a plant.

The problem of clustering regions can be stated as follows: given n 2D regions, $r_1, r_2, \ldots, r_n$, find N clusters, $c_1, c_2, \ldots, c_N$, where each region may belong to a different cluster. N is *a priori* unknown. Among the different clustering algorithms that can be found in the literature [22–27], a sequential clustering algorithm with two thresholds has been used [27], which was found to be faster than typical agglomerative clustering algorithms,

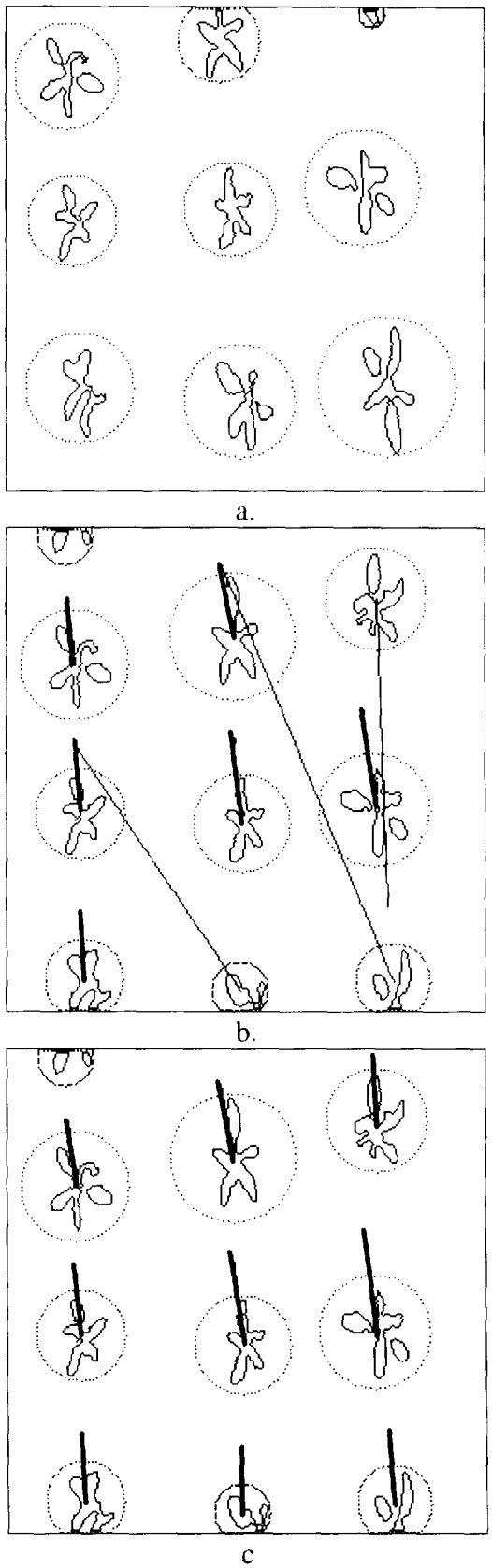

Fig. 2. Matching of plants. (a) First image of a sequence, (b) second image showing the initial matching, (c) final matching.

which are $O(n^3)$, or $O(n^2 \log n)$, but with n^2 computations of distances between points and clusters. The number of distance computations has always been less than n^2 with the images of the field that have been used here.

The distance between a pair of regions can be defined as the minimum value of the distances between any pair of points belonging to each boundary. This distance would be quite expensive to compute, and as we are clustering regions, we would have to compute the distance from one region to a set of regions, i.e. to a cluster. Another distance measurement is computed to make the process of identification fast enough; the distance computed is not exactly the distance between two regions as defined above, but it gives a sufficiently good value for the proximity between regions.

The distance between one region and one cluster (group of regions) is measured by drawing a line from the centroid of the region to the centroid of the cluster. The point belonging to the region, furthest from its centroid, is stored, and so is the point belonging to any region of the cluster, furthest from the cluster's centroid. Then, the Euclidean distance between these two points is computed. Images with the result of the clustering can be seen in Fig. 2; the clustered regions are enclosed by dotted circles.

3. Tracking strategy

Given image k at time t, and image $k+1$ at time $t+\Delta t$, the plants in image $k+1$ have to be matched with plants in image k. The matching must be consistent with some rigidity constraints, since the plants are assumed not to move, and no plant can have more than one match. Newly appearing plants in image $k+1$ are left without matching.

The tracking strategy can be divided into the following steps:

- First, a measure of similarity between two plants, i in image $k+1$, and j in image k, is computed as sim_{ij}, where $0 \leqslant sim_{ij} \leqslant 1$. $sim_{ij} = 0$ means plants i and j are completely different; $sim_{ij} = 1$ means they are completely equal. sim_{ij} is computed from the shape description of the two plants, and it will be explained later.
- Second, for each pair of consecutive images, k and $k+1$, a similarity matrix, **SM**, is built up. Let N_k be number of plants in image k, and N_{k+1} the number of plants in image $k+1$, then the matrix has N_{k+1} rows and N_k columns; element $\mathbf{SM}[i,j]$ represents the similarity between plant i in image $k+1$ ($i = 1, \ldots, N_{k+1}$) and plant j in image k ($j = 1, \ldots, N_k$). The similarity between plants is computed taking into account the previous motion parameters of the camera; each centroid of a plant in image $k-1$ is moved backwards

with the previous motion, if the centroid of plant j in image k is within a certain distance threshold of the back-moved centroid of plant i in image $k+1$, then $\mathbf{SM}[i,j]$ is set to sim_{ij}, otherwise $\mathbf{SM}[i,j]$ is set to 0. The first time the matching is computed, no previous motion exists, and no threshold for the distance is used. This distance threshold is set to 1/2 of the plant dimension, and the plant dimension is computed as (image area/typical number of plants)$^{1/2}$.

- Third, the maximum value of each row of the similarity matrix is found, and if this maximum is bigger than a similarity threshold, a potential matching between plants i and j is stored, this being the row i of the matrix and the column j where the maximum occurs. The similarity threshold used was 0.4.

This gives us a collection of initial matches. Some of them can be erroneous, as the shape description of the plants can suffer small changes when plants are appearing in, or disappearing from, the scene, or the regions that form each plant split or merge. However, the correct matches can be supposed to be more numerous than the erroneous ones. The good matches have to come from the same movement of the camera, while erroneous matches will represent movements in arbitrary directions, so good matches can be selected by clustering the possible movement that each match represents, and by keeping the cluster supported by a greater number of matches.

3.1. Selecting good matches

Since the vehicle is moving over the ground plane, with the optical axis of the camera vertical to the ground, the motion between subsequent images can be supposed to be a planar motion, with a one-angle rotation followed by a two-dimensional translation. Let us assume that plants i, in image $k+1$, and j, in image k, have been matched. Let y_i be the position of the centroid of plant i, and x_j the same for plant j; y_i and x_j are two-dimensional vectors. Then let us assume that point x_j is rotated and translated up to the matched point y_i:

$$y_i = \mathbf{R}x_j + \mathbf{T}$$

```
3DSpace
    For each potential match (i, j)
        For θ = −θlimit to θlimit in steps of θstep
            Find values of tx and ty
            Draw a point in a three-dimensional
            space with coordinates (θ, tx, ty)
        EndFor
    EndFor
End of 3DSpace
```

Fig. 3. Building a three-dimensional parameter space to select the good matches.

```
SeqClustering
    Initialize a queue to empty
    Assign point p1 to cluster c1
    k = 1          /* number of clusters so far */
    For i = 2 to n  /* process points */
        ProcessPoint(pi)
    EndFor
    /* now some points have been assigned to clusters */
    /* and others stored in the queue */

    /* process points in queue */
    While queue not empty
        For i = 1 to number of points in queue
            Take out first point in queue, p
            ProcessPoint(p)
        EndFor
        If queue has changed
            Take out first point in queue, p
            k = k + 1 /* new cluster */
            Assign point p to cluster ck
        EndIf
    EndWhile
End of SeqClustering

ProcessPoint(pi)
    Compute distances from pi to existing clusters
    Let j be number of the cluster with minimum distance
    to pi
    Let d be this minimum distance
    If d <= t1
        Assign point pi to cluster ci
    Else
        If d >= t2
            k = k + 1 /* new cluster */
            Assign point pi to cluster ck
        Else
            Add pi to the queue for later
            processing
        EndIf
    EndIf
End of ProcessPoint
```

Fig. 4. Sequential clustering algorithm with two thresholds, used to select the good matches in the parameter space.

where $\mathbf{R}$ is a 2×2 rotational matrix and $\mathbf{T}$ is a two-dimensional translation vector:

$$\mathbf{R} = \begin{pmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{pmatrix} \quad \mathbf{T} = \begin{pmatrix} t_x \\ t_y \end{pmatrix}.$$

We have two equations and three unknowns, θ, t_x and t_y. Fixing θ, t_x and t_y can be found, so we can vary θ within a range of values, and draw points in a 3D space formed by triplets (θ, t_x, t_y). This space consists of a stack of planes, one for each value given to θ, in which t_x and t_y for the corresponding θ lie (Fig. 3). Doing a clustering in this space we can find the motion supported by the greatest number of matches, which are supposed to be the good matches. The method can be thought of as a Hough transform-like technique, as it separates the good matches from the bad ones while rejecting outliers, and it give us a first guess for the camera motion parameters.

Let r_1 and r_2 be the radius of circles 1 and 2, let us assume that $r_1 \geq r_2$. Let c_1 and c_2 be the centres of circles 1 and 2.

SwapArea
  $d = \|c_1 - c_2\|$
  If $d \geq r_1 + r_2$
    $area = 0$
  Else
    If $d \leq r_1 - r_2$
      $area = \pi r_2^2$
    Else
      $q = [r_2^2 - ((r_1^2 - r_2^2 - d^2)/(2d))^2]^{1/2}$
      If $r_1^2 - r_2^2 - d^2 > 0$
        $area = \pi r_2^2 + q(r_1^2 - q^2)^{1/2} + r_1^2 \sin^{-1}(q/r_1)$
        $-q(r_2^2 - q^2)^{1/2} - r_2^2 \sin^{-1}(q/r_2) - 2dq$
      Else
        $area = q(r_1^2 - q^2)^{1/2} + r_1^2 \sin^{-1}(q/r_1)$
        $+q(r_2^2 - q^2)^{1/2} - r_2^2 \sin^{-1}(q/r_2) - 2dq$
      EndIf
    EndIf
  EndIf
End of SwapArea

Fig. 5. Intersecting area of two circles.

A sequential clustering algorithm with two thresholds has been used here, just as it was in the problem of clustering regions to find plants. To make the clustering process faster, points are ordered by the rotation angle; ordering is easily done while points are being computed. Here the distance between two points, (θ, t_x, t_y) and (θ', t'_x, t'_y), is computed as $\text{abs}(\theta - \theta') + \text{abs}(t_x - t'_x) + \text{abs}(t_y - t'_y)$ (Fig. 4).

Note that discrete and limited values have been given to the rotation angle only, allowing any value for the translation. This method permits us to constrain the values of the rotation angle. Any value can be found if θ_{limit} is set to π, but if the expected rotation between frames is going to be small, a lower value can be used. In this application a value of $\theta_{\text{limit}} = 10^\circ$ and $\theta_{\text{step}} = 2^\circ$ have been used.

The result of the selection of matches and the first guess for the motion parameters can be seen in Fig. 2(b), an image which follows the one in Fig. 2(a). Lines represent potential matches, starting from the centroid of a plant in image $k+1$ and ending in the centroid of the corresponding matched plant in image k. Thick lines represent the good matches found, and thin lines the bad matches.

3.2. Finding the rest of the matches

Now that the good matches have been separated from the bad ones, better values for the motion parameters can be found by minimisation. This will be explained later, but let us assume that we already know the motion parameters. Then we can move back the plants that have not yet been matched, overlap these plants in the previous image and find their matches. To make the process fast enough for real time purposes, each plant is only represented by its enclosing circle; the centre and radius of this circle was found when the shape description was computed. Each circle of a still not matched plant in image $k+1$ is back-rotated and translated, and the intersecting area with every circle of still not matched plants in image k is computed. The intersecting area of two circles is computed as described in Fig. 5.

The maximum intersecting area of each plant in image $k-1$ is stored in a match table. Then, matches are selected from it by taking out the biggest value at each time. If one plant in image $k+1$ is found to be matched with an already matched plant in image k, the intersecting area that has produced this matching is discarded, and another maximum intersecting area is found for that plant in image $k+1$. If one plant in image $k+1$ has a maximum intersecting area of zero, it is left without any match, which means that it is a newly appearing plant in the image. The final matching of plants can be seen in Fig. 2(c).

3.3. Shape description

The matching process explained so far is based on a measure of similarity between plants, computed from their shape description. After the identification of plants, each consists of one or more 2D regions. In subsequent images, regions may split or merge, but each plant remains with a similar shape. The shape description must have the following properties: (1) It must handle the splitting and merging of regions; (2) it must be invariant to rotations and translations; (3) it must be sensitive to scale changes.

A shape description invariant to scaling is not suitable in this problem, because the camera is always at the same distance from the ground, and this scale sensitivity will help us to discriminate between different plants. The shape description adopted is like taking a signature of the plant [28], and a starting angle to make it invariant to rotations. For each plant, a circle enclosing the plant is computed, and then at fixed angle steps, a line is drawn from this circle to the centroid of the plant. The first point at which the line crosses any of the boundaries of the plant is kept, and the distance between this point and the centroid is recorded. Thus, we have a one-dimensional signal for each plant, representing the distance to the centroid of the furthest point of the plant, at regular angles.

The starting angle is computed as the orientation in terms of the minimum moment of inertia of the regions representing the plant. First and second order geometrical moments are computed from the boundary for every region of the plant [29–31]. Then moments of each region are added to compute the first and second order moments of the plant. Geometrical moments of order pq, m_{pq}, centred moments, μ_{pq} and the centroid (c_x, c_y) are defined as:

$$m_{pq} = \iint_{\text{plant area}} x^p y^q \, \mathrm{d}x\mathrm{d}y$$

$$\mu_{pq} = \iint_{\text{plant area}} (x - c_x)^p (y - c_y)^q \, \mathrm{d}x\mathrm{d}y$$

$$c_x = \frac{m_{10}}{m_{00}}; \quad c_y = \frac{m_{01}}{m_{00}}.$$

The maximum and minimum moments of inertia and their corresponding angles are computed as:

$$I_{\min} = \frac{\mu_{20} + \mu_{02} - \sqrt{(2\mu_{11})^2 + (\mu_{20} - \mu_{02})^2}}{2}$$

$$I_{\max} = \frac{\mu_{20} + \mu_{02} + \sqrt{(2\mu_{11})^2 + (\mu_{20} - \mu_{02})^2}}{2}$$

$$\varphi_{\min} = \frac{1}{2}\tan^{-1}\frac{2\mu_{11}}{\mu_{20} - \mu_{02}} \qquad \varphi_{\max} = \varphi_{\min} + \frac{\pi}{2}.$$

Only the computation of $\varphi_{\min}$ is required for our shape description.

The shape description consists of $\varphi_{\min}$, and a p-dimensional vector, $\mathbf{sd} = (sd[0], sd[1], \ldots, sd[p-1])^{\mathrm{T}}$. Each position can only take positive real values. If φ_s is the angle step, then $sd[l]$ is the Euclidean distance between the furthest point to the centroid of the plant at angle $l\varphi_s$, and the dimension of the vector is $p = 2\pi/\varphi_s$. The vector is considered circular, which means that position $l + np$ is considered to be the same position l $(n = 1, 2, \ldots)$. The starting position in the vector, st, is the nearest integer to $\varphi_{\min}/\varphi_s$.

With these considerations in mind, the similarity between two plants, i and j, is computed as $sim_{ij} = \exp(-a_{ij}/\lambda_{ij})$, where:

$$a_{ij} = \sum_{l=0..p-1} \{\mathrm{abs}(sd_i[l + st_i] - sd_j[l + st_j])\}$$

$$\lambda_{ij} = \min\left(\sum_{l=0..p-1} \{sd_i[l]\}, \sum_{l=0..p-1} \{sd_j[l]\} \right)$$

$$\text{and} \quad a_{ij} \geqslant 0, \lambda_{ij} \geqslant 0, 0 \leqslant sim_{ij} \leqslant 1.$$

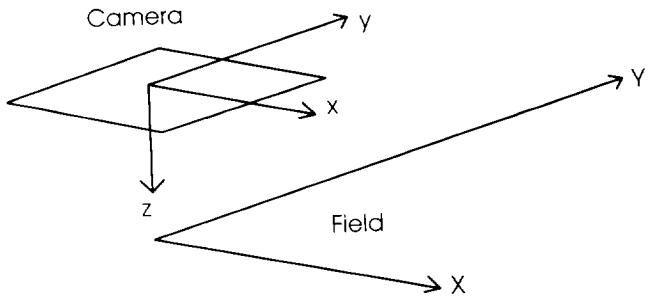

Fig. 6. Geometry of the camera and the field.

4. Finding the vehicle motion parameters

The objective is to find the translation and rotation of the vehicle in any image k, with respect to the first image of the sequence, where the origin of the field coordinates is set. These rotations and translations can be found from the previous position of the vehicle, and from the motion from image $k-1$ to image k.

Let $\mathbf{R}_k$, $\mathbf{T}_k$, be the rotation and translation in image k, with respect to the first image of the sequence. Let $\mathbf{r}_k$, $\mathbf{t}_k$, be the rotation and translation from image $k-1$ to image k, computed from the movement seen in the image, which is opposite to the movement of the vehicle. $\mathbf{R}_k$ and $\mathbf{r}_k$ are two-dimensional rotation matrices, $\mathbf{T}_k$ and $\mathbf{t}_k$ are two-dimensional translation vectors. Then, $\mathbf{R}_k$ and $\mathbf{T}_k$ can be found as:

$$\mathbf{R}_k = \mathbf{R}_{k-1}\,\mathbf{r}_k^{-1}; \qquad \mathbf{R}_0 = \mathbf{I}$$

$$\mathbf{T}_k = \mathbf{T}_{k-1} - \mathbf{R}_k\,\mathbf{t}_k; \quad \mathbf{T}_0 = \mathbf{0}$$

The motion parameters in subsequent images, $\mathbf{r}_k$ and $\mathbf{t}_k$, are computed using a set of point correspondences between the two images. These feature points are the centres of the plants that have been matched in the tracking stage. The geometry of the problem is as follows: the vehicle is moving over the ground, so the movement approximately consists of a one-angle rotation followed by a 2D translation. The camera is mounted with its vertical axis perpendicular to the ground plane, so the rotation is made around the z axis of the camera (Fig. 6). With this configuration, the projection of the centres of the plants in the image should undergo a rotation followed by a translation in the image plane. But if the

camera is not exactly perpendicular to the ground, this will not be true, and if a method that only considers a 2D movement in the image plane is used, small errors in the rotation angle could be obtained. These errors will accumulate from image to image, resulting in an erroneous trajectory of the vehicle after a few images, and in an erroneous map of the field. A method that considers 3D motion is used to overcome these problems, including some constraints to introduce our knowledge about the movement; that is, the vehicle is moving over the ground, and the camera has its vertical axis nearly perpendicular to the ground.

Let $M = \{\mathbf{M}_1, \mathbf{M}_2, \ldots, \mathbf{M}_n\}$ be a set of points in the 3D world. These points are seen under different perspectives in images k and $k-1$, so let $M_k = \{\mathbf{M}_{1,k}, \mathbf{M}_{2,k}, \ldots, \mathbf{M}_{n,k}\}$ be the point M given in camera coordinates at image k, and $M_{k-1} = \{\mathbf{M}_{1,k-1}, \mathbf{M}_{2,k-1}, \ldots, \mathbf{M}_{n,k-1}\}$ be the same points given in camera coordinates at image $k-1$, $(\mathbf{M}_{i,k} = (X_{i,k}, Y_{i,k}, Z_{i,k})^{\mathrm{T}})$. Let $m_k = \{\mathbf{m}_{1,k}, \mathbf{m}_{2,k}, \ldots, \mathbf{m}_{n,k}\}$ be the projections on the image plane of the points M_k, and let $m_{k-1} = \{\mathbf{m}_{1,k-1}, \mathbf{m}_{2,k-1}, \ldots, \mathbf{m}_{n,k-1}\}$ be the projections of the points M_{k-1}. Using homogeneous coordinates, $\mathbf{m}_{i,k} = (x_{i,k}, y_{i,k}, 1)^{\mathrm{T}} = (wx_{i,k}, wy_{i,k}, w)^{\mathrm{T}}$. Since points M are the centres of the plants that have been matched in images k and $k-1$, we can assume that they belong to the same plane, a plane that is viewed from two different positions of the camera in images k and $k-1$. It can be shown [31] that the projected points are related by a collineation matrix $\mathbf{m}_{i,k} = \mathbf{A}\,\mathbf{m}_{i,k-1}$ when these points lie in a plane in the three-dimensional space. $\mathbf{A}$ is defined up to a scale factor, since points $\mathbf{m}_{i,k}$ and $\mathbf{m}_{i,k-1}$ are given in homogeneous coordinates.

From image $k-1$ to image k, the camera undergoes a rotation followed by a translation, so $\mathbf{M}_{i,k} = \mathbf{R}_3\,\mathbf{M}_{i,k-1} + \mathbf{T}_3$. $\mathbf{R}_3$ is a 3×3 rotation matrix and $\mathbf{T}_3$ is a 3D translation vector:

$$\mathbf{R}_3 = \begin{pmatrix} \cos\varphi_z \cos\varphi_y + \sin\varphi_z \sin\varphi_x \sin\varphi_y & -\sin\varphi_z \cos\varphi_x & \sin\varphi_z \sin\varphi_x \cos\varphi_y - \cos\varphi_z \sin\varphi_y \\ \sin\varphi_z \cos\varphi_y - \cos\varphi_z \sin\varphi_x \sin\varphi_y & \cos\varphi_z \cos\varphi_x & -\cos\varphi_z \sin\varphi_x \cos\varphi_y - \sin\varphi_z \sin\varphi_y \\ \cos\varphi_x \sin\varphi_y & \sin\varphi_x & \cos\varphi_x \cos\varphi_y \end{pmatrix} \quad \mathbf{T}_3 = \begin{pmatrix} t_x \\ t_y \\ t_z \end{pmatrix}.$$

φ_x, φ_y and φ_z are the angles of rotation around axes x, y and z.

Points M_{k-1} belong to a plane determined by its normal, $\mathbf{n}$, and its distance to the centre of the camera, d, in camera coordinates at image $k-1$. So:

$$\mathbf{n}^{\mathrm{T}}\,\mathbf{M}_{i,k-1} = d.$$

With these two equations, we have that:

$$\mathbf{M}_{i,k} = \mathbf{R}_3\,\mathbf{M}_{i,k-1} + ((\mathbf{T}_3\,\mathbf{n}^{\mathrm{T}})/d)\,\mathbf{M}_{i,k-1}$$

$$\mathbf{M}_{i,k} = (\mathbf{R}_3 + (\mathbf{T}_3\,\mathbf{n}^{\mathrm{T}})/d)\,\mathbf{M}_{i,k-1}.$$

Since the coordinates of $\mathbf{M}_{i,k}$ and $\mathbf{M}_{i,k-1}$ are given in camera coordinates, they are the projective homogeneous coordinates of $\mathbf{m}_{i,k}$ and $\mathbf{m}_{i,k-1}$; so, $\mathbf{A} = \mathbf{R}_3 + (\mathbf{T}_3\,\mathbf{n}^{\mathrm{T}})/d$. Since $\mathbf{A}$ is defined up to a scale factor, multiplying this equation by d, we have $\mathbf{A} = d\,\mathbf{R}_3 + \mathbf{T}_3\,\mathbf{n}^{\mathrm{T}}$. $\mathbf{t}_k$ and d are also given up to a scale factor; this is known as the speed-scale ambiguity.

Now, applying some knowledge about the movement, we can introduce these constraints to the method:

- The camera is always at the same distance from the ground, so we can choose any constant value for d. We make $d = 1$.
- The vehicle is moving over the ground, so the z component of the translation will be zero.
- The optical axis of the camera is perpendicular to the ground, so the rotation is around the z axis.

With these constraints, $\mathbf{R}_3$, $\mathbf{T}_3$ and the collineation matrix, $\mathbf{A}$, become:

$$\mathbf{R}_3 = \begin{pmatrix} \cos\varphi_z & -\sin\varphi_z & 0 \\ \sin\varphi_z & \cos\varphi_z & 0 \\ 0 & 0 & 1 \end{pmatrix} \quad \mathbf{T}_3 = \begin{pmatrix} t_x \\ t_y \\ 0 \end{pmatrix} \quad \mathbf{A} = \begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ 0 & 0 & 1 \end{pmatrix}.$$

$\mathbf{R}_3$, $\mathbf{T}_3$ and $\mathbf{n}$ are computed from $\mathbf{A}$ using a method based on the singular value decomposition of matrix $\mathbf{A}$ [32–35]. In the general case, this method can give a maximum of four solutions, two of which can be rejected as they represent the same solutions as the other two, but with a change of sign for $\mathbf{T}_3$ and $\mathbf{n}$. From the other two, the one with a rotation, $\mathbf{R3}$, closer to the type of matrix expected, $\varphi_x \approx 0$ and $\varphi_y \approx 0$, is selected.

To compute matrix $\mathbf{A}$, six unknowns have to be found; this means that at least three non-collinear point correspondences are needed to recover this type of motion. Usually, we will have more than three point correspondences, so the six unknowns have to be found by least-squares minimisation. The method is:

- Arrange the unknowns in a vector, $\mathbf{a} = (a_{11}, a_{12}, a_{13}, a_{21}, a_{22}, a_{23})$.
- Write the $2n$ equations $\mathbf{m}_{i,k} = \mathbf{A}\,\mathbf{m}_{i,k-1}$ $(i = 1, 2, \ldots, n)$, as $\mathbf{Ba} = \mathbf{b}$, where $\mathbf{B}$ is a $2n \times 6$ matrix, and $\mathbf{b}$ is a $2n$ by 1 vector.

- Apply a least-squares minimisation to solve a linear system with more equations than unknowns. This can be done by singular value decomposition of matrix **B**.

From the 3D rotation and translation found, $\mathbf{R}_3$ and $\mathbf{T}_3$, a 2D rotation and translation from image k to image $k+1$ have to be found, $\mathbf{r}_k$ and $\mathbf{t}_k$. $\mathbf{r}_k$ is computed from the angle of rotation around the z axis in $\mathbf{R}_3$, φ_z; and $\mathbf{t}_k$ is computed from the x and y components of $\mathbf{T}_3$, t_x and t_y. $\mathbf{r}_k$ and $\mathbf{t}_k$ are used to update the position of the vehicle in the field, as was explained before.

As pointed out before, the method used to find the collineation matrix **A** has some differences with the general method [32–35]. In general, it is supposed that the motion is three-dimensional, without any constraint, so **A** has eight degrees of freedom (a_{33} is set to one because **A** is defined up to a scale factor). At least four point correspondences have to be used to find the eight unknowns. With our method, only six unknowns have to be found from at least three point correspondences. Fixing $a_{31}=0$ and $a_{32}=0$ has the advantage that the motion parameters computed from **A**, $\mathbf{R}_3$, $\mathbf{T}_3$ and **n** are less noise dependent. Using the general method with our data, a_{31} and a_{32} are expected to have a value of nearly zero. The values found have been of the order of 10^{-4}. These values have a negative influence in the subsequent computation of $\mathbf{R}_3$, $\mathbf{T}_3$ and **n**, and the results obtained do not correspond to the motion expected. The two methods have been compared, and the results of applying our method have been much better than the results using the general method.

5. Computing the map of the field

A map of the field consists of a set of positions, in world coordinates, where the plants lie. World coordinates are defined in the first image of the sequence, coinciding with the x and y axes of the camera coordinates. When a plant is seen in image k, we know its position in camera coordinates at image k; the world coordinates of the plant can be found knowing the position and orientation of the vehicle, $\mathbf{R}_k$ and $\mathbf{T}_k$. Let $\mathbf{c}=(x,y)^{\mathrm{T}}$ be the centre of a plant in image coordinates at image k, and let $\mathbf{C}=(X,Y)^{\mathrm{T}}$ be the same centre in world coordinates; then $\mathbf{C}=\mathbf{c}\,\mathbf{R}_k+\mathbf{T}_k$.

Each plant is tracked from image to image, so we have a number of observations of each plant. The real position of the plant in the field is computed from its observations, using a Kalman filter. The Kalman filter is a minimisation technique used to find the best value of a state vector in Gaussian linear systems. It is based on a noisy linear model of a dynamic system, and on a noisy measurement model.

Let $\mathbf{x}_k$ be a n-dimensional state vector. From state $\mathbf{x}_{k-1}$, the system goes to state $\mathbf{x}_k$ by applying matrix Φ_k, but the new state is combined with noise from a random Gaussian n-dimensional vector αk, with **0** mean and covariance matrix Λ_k. Let $\mathbf{y}_k$ be an m-dimensional measurement vector. From state $\mathbf{x}_k$, a measurement is obtained by applying matrix $\mathbf{H}_k$, but the observation is corrupted again with the Gaussian noise vector β_k, with **0** mean and covariance matrix $\mathbf{B}_k$. From $k-1$ observations, a prediction of the state vector at instant k is made, $\mathbf{x}_{k|k-1}$. Then, this prediction is updated with the Kalman gain matrix, $\mathbf{K}_k$, to find the best prediction at instant k, $\mathbf{x}_{k\ k}$. The covariance matrix of the state vector, $\mathbf{P}_K$, is predicted in the same way. The initial conditions are $\mathbf{x}_{0|0}=E[\mathbf{x}_0]$ and $\mathbf{P}_0=\mathrm{cov}[\mathbf{x}_0]$. Random variables α_k and β_k are supposed to be uncorrelated, $E[\alpha_k,\beta_k^{\mathrm{T}}]=0$.

System model:

$$\mathbf{x}_k=\Phi_{k-1}\mathbf{x}_{k-1}+\alpha_k \quad ;\alpha_k\in N(\mathbf{0},\Lambda_k)$$

Measurement model:

$$\mathbf{y}_k=\mathbf{H}_k\mathbf{x}_k+\beta_k \quad ;\beta_k\in N(0,\mathbf{B}_k)$$

Initial state:

$$E[\mathbf{x}_0]=\mathbf{x}_{0|0} \quad ;\mathrm{cov}[\mathbf{x}_0]=\mathbf{P}_0$$

Prediction at time instant $k-1$:

$$\mathbf{x}_{k|k-1}=\Phi_{k-1}\mathbf{x}_{k-1|k-1}$$

$$\mathbf{P}_{k|k-1}=\Phi_{k-1}\mathbf{P}_{k-1}\Phi_{k-1}^{\mathrm{T}}+\Lambda_K$$

Prediction at time instant k:

$$\mathbf{K}_k=\mathbf{P}_{k|k-1}\mathbf{H}_k^{\mathrm{T}}[\mathbf{H}_k\mathbf{P}_{k|k-1}\mathbf{H}_k^{\mathrm{T}}+\mathbf{B}_k]^{-1}$$

$$\mathbf{x}_{k|k}=\mathbf{x}_{k|k-1}+\mathbf{K}_k[\mathbf{y}_k-\mathbf{H}_k\mathbf{x}_{k|k-1}]$$

$$\mathbf{P}_{k|k}=[\mathbf{I}-\mathbf{K}_k\mathbf{H}_k]\mathbf{P}_{k|k-1}$$

5.1. Kalman filter design

In this case, we are interested in finding the real position of a plant in the field, which is unknown, from a set of measurements of the position of this plant as it appears in subsequent images. The state vector is the real position of the plant in the field, $\mathbf{x}_k=(X,Y)^{\mathrm{T}}$. Since this position is given in world coordinates, the state vector will not change, so $\Phi_k=\mathbf{I}$. The measurements are the positions, also in world coordinates, of the plant in different images, $\mathbf{y}_k=(x_{\mathrm{meas}},y_{\mathrm{meas}})^{\mathrm{T}}$, and $\mathbf{H}_k=\mathbf{I}$. $\mathbf{x}_{0|0}$ is set as the first observation of the plant. The covariance matrices are defined by assuming that the x and y coordinates of the initial state, αk and β_k, are uncorrelated, and that their variance is the same, so:

$$\mathbf{P}_0=\sigma_x^2\mathbf{I};\quad \Lambda_k=\sigma_\alpha^2\mathbf{I};\quad \mathbf{B}_k=\sigma_\beta^2\mathbf{I}.$$

The values used for σ_x^2, δ_α^2 and σ_β^2 are related to the

size of the plants; here we have made $\sigma_x^2 = \sigma_\alpha^2 = \sigma_\beta^2 =$ (10% of the plant size)^2.

The simplicity of this Kalman filter implementation also allows a fast real-time performance. Recall that the filter is applied to each plant in every image. There is no need for an expensive matrix inversion, since matrix $[\mathbf{H}_k \mathbf{P}_{k|k-1} \mathbf{H}_k^T + \mathbf{B}_k]$ is a diagonal matrix.

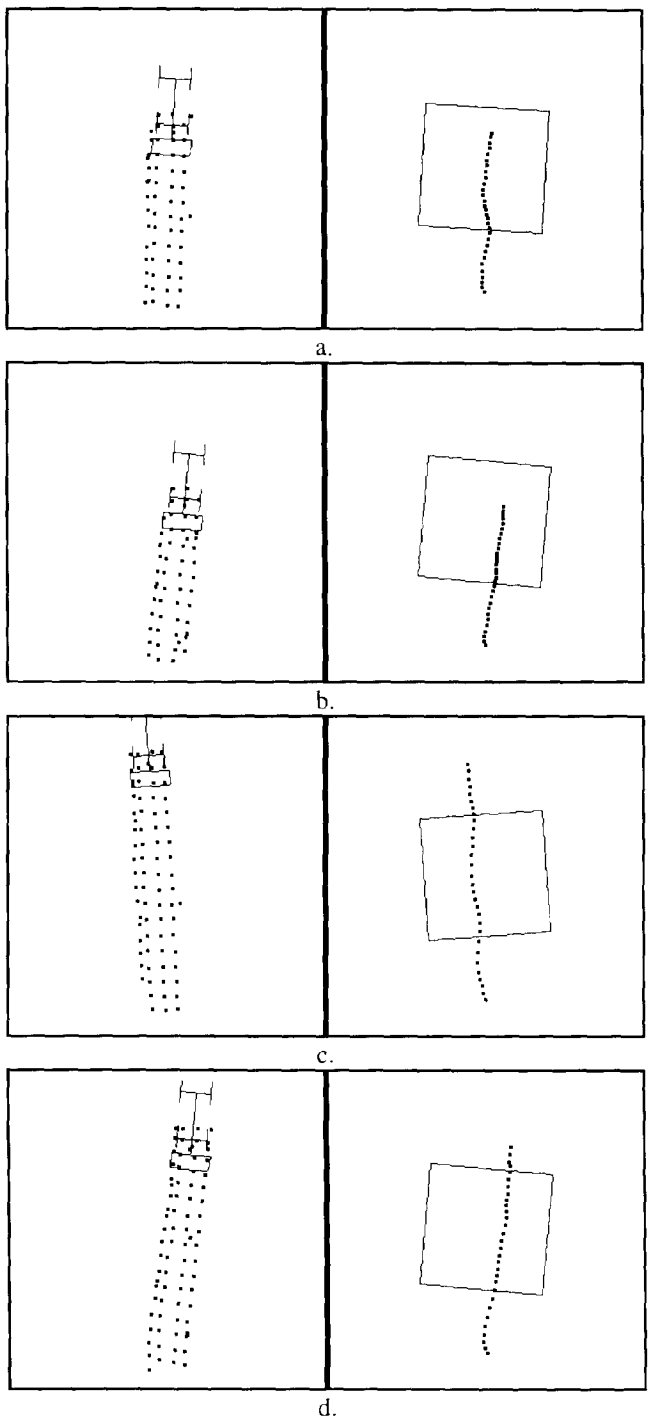

Fig. 7. Recovered motion and map of the field, zigzag movement.

6. Experimental results

The methods described so far have been tested with outdoor images taken under natural conditions. Two type of experiments were performed in the experimental fields at the Silsoe Research Institute (Beds., UK).

In the first experiment, a camera was mounted on the rear of a manually driven agricultural vehicle, with the optical axis of the camera perpendicular to the ground plane, although this orientation cannot be exact, since the ground is irregular (a crop field), the vehicle moves, and the camera suffers vibrations. The vehicle executed a zigzag movement along the field. Some sequences of 30 images, with different speeds and trajectories, were taken. After the segmentation, the regions with class 'plant' were used for the tracking system.

Fig. 7 represents the trajectory of the vehicle and the map of the field, recovered after processing each sequence of 30 images. On the left side of the figures, the map of the field can be seen, along with the final position of the vehicle. On the right side can be seen the trajectory performed by the vehicle, and the orientation of the camera in the final image, with respect to the first image. These sequences were taken in a field with the plants arranged in straight rows and, as can be seen in the figures, the plants also appear arranged in straight rows, so we can say that the structure of the field has been recovered quite accurately.

The frame rate was 140 ms, representing 4.2 s for a sequence of 30 images. Figs. 6(a) and (b) show two sequences with a vehicle speed of 1.5 metres/s that has travelled a distance of 6.3 metres. Figs. 6(c) and (d) show two sequences with a speed of 2 metres/s travelled distance of 8.4 metres. The row spacing was 60 cm and within each row plants are separated 60 cm apart. Fig. 1 shows the first two images of the sequence in Fig. 6(c).

To check the validity of the algorithms in situations with a strong component of rotation in the movement, the second experiment consisted of mounting a camera on a vertical bar, with its optical axis vertical to the ground, and to rotate the camera around the bar. Fig. 8 shows the recovered motion of the camera for a sequence of 20 images. The translation that can be seen corresponds to the centre of the camera, which rotates around the vertical bar; the final orientation of the camera is represented by the square superimposed over the trajectory.

Off-line time measurements have been done using a HP9000/730 workstation. The processing time per image is slightly different, and depends on the number of boundaries and size. The mean time for a sequence of 30 images is 77 ms; the maximum processing time has been 110 ms and the standard deviation was 17.8 ms. This time includes the computations on the boundaries,

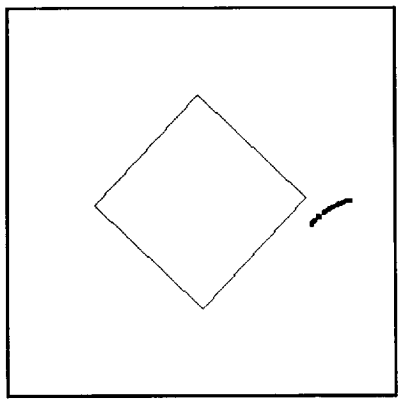

Fig. 8. Recovered rotational movement.

motion parameters and Kalman filter, but excludes the image segmentation and the chain encoding of the boundaries. The reason is that this can be done with dedicated hardware [36].

Real-time implementation is straightforward. The segmentation process can be done by input look-up tables at frame rate, and the chain coding can also be done at frame rate using dedicated hardware which has been already developed at the Silsoe Research Institute [36]. The shape description and Kalman filter applied to each plant can be parallelized by dedicating different processors to work with different plants. The whole system can be implemented on a transputer array, thus allowing simple interconnection, using transputer links, with the control system of an autonomous vehicle developed at the Silsoe Research Institute.

7. Main breakpoints of the approach

This approach is based on the following assumptions:

- illumination is good enough to segment the images;
- plants are quite separated from each other so they can be identified by the clustering of regions;
- the distance travelled from frame to frame is low enough for a complete plant to appear at least in two consecutive frames.

Since this work stresses algorithms applied to segmented images, and work related to the segmentation in plants and weeds [17] has been reported, we address the reader to those other works for a more detailed discussion on the effect of the illumination in the segmentation.

The tracking of plants is based on their identification, and that is done by clustering the regions that belong to the same plant. It is assumed that they are quite far from any region of another plant. When the plants grow up and become bigger, these distances are of the same order the identification fails. Up to two month old cabbages have been successfully identified by this method.

Fixing the frame rate and the plant separation, the distance that a plant moves in two consecutive images is related to the vehicle speed. With the size of the field of view used, plants would begin not to appear complete in two consecutive frames at a speed of 4 metres/s. In the experiments, only tests until 2 metres/s have been made.

8. Conclusions

A method to recover a map of a crop field from a sequence of images using structure from motion techniques has been presented. It is based on identifying and tracking the plants that appear in the scene. Its application is intended to be the automation of some agricultural tasks, by mounting a vision system and a treating device on the back of an agricultural vehicle. The motion parameters of the camera are recovered, and they can be used by the guidance system of an autonomous vehicle, although the system could be used as a trolley towed by a manually driven vehicle as well. The output of the vision system would be the input to a task planning module that will decide to apply chemical treatment to the plants.

Clustering and shape description methods for objects that consist of more than one region have been presented, and they handle the splitting and merging of regions in subsequent images. A Hough transform-like technique, followed by a clustering in a three-dimensional space, has been designed to select matches consistent with the rigidity of the scene. This method is applied to the initial matching to separate the good matches, and then a minimisation criterion can be used to find the motion parameters, with the advantage of knowing that the data used is very likely to be correct.

The motion parameters of the camera are recovered from a set of point correspondences, the centres of the plants. The method used assumes that all the points lie in a plane, and includes some constraints in the motion to compute the collineation matrix that relates the two sets in a pair of consecutive images. The different observations of the plant are integrated by applying a Kalman filter to each one, in order to place the plants in the field.

Although just a local map between the camera field of view and the treating device is necessary for the purpose of this application, by recording a full map for the complete travelling of the vehicle, it can be seen if the row structure of the field has been recovered successfully, even in cases of zigzag movement, thus providing a degree of confidence in the recovered motion parameters which could be eventually used in the guidance of an autonomous vehicle.

Acknowledgements

This work has been partially supported by the BBSRC, UK, and a grant from Fundacio Caixa Castello,

Spain. The authors wish to thank Dr Inesta for his comments on earlier versions of this paper.

References

[1] R. Wallace, A. Stenz, C. Thorpe, H. Moravec, W. Whittaker and T. Kanade, First results in robot road following, Proc. 9th Int. Conf. Artificial Intelligence, vol. 2, Los Angeles, CA, 1985, pp. 1089–1095.

[2] S.P. Liou and R.C. Jain, Road following using vanishing points, Proc. IEEE Conf. Computer Vision and Pattern Recognition, 1986, pp. 41–46.

[3] C. Thorpe, M.H. Herbert, T. Kanade and S.A. Shafer, Vision and navigation for the Carnegie-Mellon Navilab, IEEE Trans. Patt. Analysis and Machine Intell., 10 (3) (1988) 362–373.

[4] M. Turk, K.D. Morgenthaler, K.D. Gremban and M. Marra, A vision system for autonomous land vehicle navigation, IEEE Trans. Patt. Analysis and Machine Intell., 10 (3) (1988) 342–360.

[5] N. Ayache and O. Faugueras, Building, registering, and fusing noisy visual maps, Proc. 1st Int. Conf. Computer Vision, 1987, pp. 73–82.

[6] S. Tsuji and J.Y. Zheng, Visual path planning, Proc. 10th Int. Joint Conf. Artificial Intelligence, vol. 2, 1987, pp. 1127–1130.

[7] Y.F. Wang, N. Karandikar and K. Aggarwal, Analysis of video image sequences using point and line correspondences, Pattern Recognition, 24 (1993) 1065–1084.

[8] Y. Liu and T.S. Huang, Three dimensional motion determination from real scene images using straight line correspondences, Pattern Recognition, 25 (1993) 617–639.

[9] T.N. Tan, K.D. Baker and G.D. Sullivan, 3D structure and motion estimation from 2D image sequences, Image & Vision Computing, 11 (1993) 203–210.

[10] Z. Zhang and O. Faugueras, Determining motion from 3D line segment matches: a comparative study, Image & Vision Computing, 9 (1991) 10–19.

[11] E.D. Dickmanns and V. Graefe, Dynamic monocular machine vision, Machine Vision and Applic., 1 (1988) 223–240.

[12] E.D. Dickmanns and V. Graefe, Applications of dynamic monocular machine vision, Machine Vision and Applic., 1 (1988) 241–261.

[13] L. Matthies, T. Kanade and R. Szeliski, Kalman filter-based algorithms for estimating depth from image sequences, Int. J. Computer Vision, 3 (1989) 209–236.

[14] G. Sandini and M. Tistarelli, Active tracking strategy for monocular depth over multiple frames, IEEE Trans. Patt. Analysis and Machine Intell., 12 (1) (1990) 13–27.

[15] N. Cui, J. Weng and P. Cohen, Extended structure and motion analysis from monocular image sequences, Proc. 3rd Int. Conf. Computer Vision, 1990, pp. 222–229.

[16] J. Santos-Victor and J. Senteiro, Generation of 3D dense depth maps by dynamic vision, Proc. Br. Machine Vision Conf., vol 1, 1992, pp. 129–138.

[17] R. Brivot and J.A. Marchant, Segmentation of plants and weeds using infrared images, Acta Horticulturae, 1995 (in press).

[18] C.G. Harris and J.M. Pike, 3D positional integration from image sequences, Proc. 3rd Alvey Vision Conf., 1987, pp. 233–236.

[19] M. Stephens and C.G. Harris, 3D wire-frame integration from image sequences, Proc. 4th Alvey Vision Conf., 1988, pp. 159–165.

[20] R. Evans, Kalman filtering and pose estimates in applications of the RAPID Video Rate Tracker, Proc. Br. Machine Vision Conf., 1990, pp. 79–84.

[21] Z. Zang, Strategies for tracking tokens in a cluttered scene, Proc. Br. Machine Vision Conf., vol 1, 1993, pp. 205–216.

[22] J. Boberg and T. Salakosi, General formulation and evaluation of agglomerative clustering methods with metric and non-metric distances, Pattern Recognition, 26 (1993) 1395–1406.

[23] R.O. Duda and P.E. Hart, Pattern Classification and Scene Analysis, Wiley, New York, 1973.

[24] K. Hattori and Y. Torii, Effective algorithms for the nearest neighbour method in the clustering problem, Pattern Recognition, 26 (1993) 741–746.

[25] T. Kurita, An efficient agglomerative clustering algorithm using a heap, Pattern Recognition, 24 (1991) 205–209.

[26] R. Dubes and A.K. Jain, Clustering methodologies in exploratory data analysis, Adven. Computer, 19 (1980) 113.

[27] P. Trahanias and E. Skordalaskis, An efficient sequential clustering method, Pattern Recognition, 22 (1989) 449–453.

[28] Y. He and A Kundu, 2-D shape classification using hidden Markov model, IEEE Trans. Patt. Analysis and Machine Intell., 13 (1991) 1171–1184.

[29] B.C. Li, A new computation of geometric moments, Pattern Recognition, 26 (1993) 109–113.

[30] X.Y. Jiang and H. Bunke, Simple and fast computation of moments, Pattern Recognition, 24 (1991) 801–806.

[31] J.G. Leu, Computing a shape's moments from its boundary, Pattern Recognition, 24 (1991) 949–957.

[32] O. Faugueras, Three-Dimensional Computer Vision, a Geometric Viewpoint, MIT Press, Cambridge, MA, 1993.

[33] R.Y. Tsai and T.S. Huang, Estimating 3-D motion parameters of a rigid planar patch, IEEE Trans. Acoustics, Speech and Signal Processing, 29 (7) (1981) 1147–1152.

[34] R.Y. Tsai, T.S. Huang and W.L. Zhu, Estimating 3-D motion parameters of a rigid planar patch, II: Singular value decomposition, IEEE Trans. Acoustics, Speech and Signal Processing, 30 (4) (1982) 525–534.

[35] R.Y. Tsai and T.S. Hunag, Estimating 3-D motion parameters of a rigid planar patch, III: Finite point correspondences and the three-view problem, IEEE Trans. Acoustics, Speech and Signal Processing, 32 (2) (1984) 213–220.

[36] J.A. Marchant and R.D. Tillet, Software and transputer system design for high speed grading of agricultural produce, Mechatronics, 4 (3) (1994) 281–293.