The development and evaluation of computer vision algorithms for the control of an autonomous horticultural vehicle

J B Southall

A dissertation submitted in partial fulfillment

of the requirements for the degree of

Doctor of Philosophy

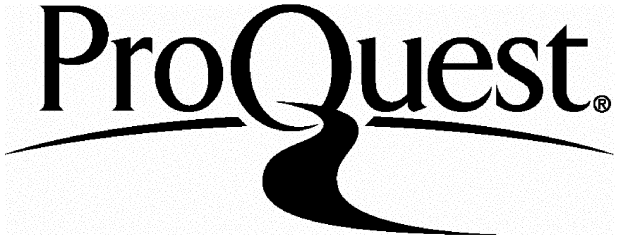
of the

University of London.

Department of Computer Science

University College London

1999



Abstract

Economic and environmental pressures have led to a demand for reduced chemical use in crop production. In response to this, *precision agriculture* techniques have been developed that aim to increase the efficiency of farming operations by more targeted application of chemical treatment. The concept of plant scale husbandry (PSH) has emerged as the logical extreme of precision techniques, where crop and weed plants are treated on an individual basis. To investigate the feasibility of PSH, an autonomous horticultural vehicle has been developed at the Silsoe Research Institute.

This thesis describes the development of computer vision algorithms for the experimental vehicle which aim to aid navigation in the field and also allow differential treatment of crop and weed. The algorithm, based upon an extended Kalman filter, exploits the semi-structured nature of the field environment in which the vehicle operates, namely the grid pattern formed by the crop planting. By tracking this grid pattern in the images captured by the vehicle's camera as it traverses the field, it is possible to extract information to aid vehicle navigation, such as bearing and offset from the grid of plants. The grid structure can also act as a cue for crop/weed discrimination on the basis of plant position on the ground plane. In addition to tracking the grid pattern, the Kalman filter also estimates the mean distances between the rows of lines and plants in the grid, to cater for variations in the planting procedure.

Experiments are described which test the localisation accuracy of the algorithms in off-line trials with data captured from the vehicle's camera, and on-line in both a simplified test-bed environment and the field. It is found that the algorithms allow safe navigation along the rows of crop. Further experiments demonstrate the crop/weed discrimination performance of the algorithm, both off-line and on-line in a crop treatment experiment performed in the field where all of the crop plants are correctly targeted and no weeds are mistakenly treated.

Acknowledgements

Sincere thanks must go to my two supervisors, Bernard Buxton at UCL and John Marchant at the Silsoe Research Institute, for their advice, encouragement and enthusiasm that has made the last three years as enjoyable as they have been educational. Danny Alexander read much of this thesis in draft form, and I thank him for his comments and pertinent questions. Simon Arridge and Søren-Aksel Sørensen at UCL, and Andy Frost and Robin Tillett at Silsoe have participated actively in the various committees set up to monitor my progress over the last three years, and their comments have been most helpful.

This project would not have been possible without the contributions of a number of people at the Silsoe Research Institute. Tony Hague has been responsible for the design and development of the sensing and estimation systems on the autonomous vehicle that the work in this thesis is centred upon, and it is safe to say that without his efforts the on-line demonstrations presented here would not have been possible. I also thank Tony for many conversations on the practical use of Kalman filters, and much of my understanding of the subject is down to him. Ian Jeffs and Chris Slatcher supervised the growth of the cauliflowers used in experiments throughout this thesis, and Bob Wardell took the photographs of the treated crop seen in chapter 8. His patience in assembling the mosaics by hand is much appreciated. Simon Miles, John Butler and John Richards provided much assistance in putting together all manner of bits and pieces of experimental equipment.

The members of the image analysis group at Silsoe made me particularly welcome during my stay in the lab, and I thank them all for their friendship, advice and library code. Dickson Chan and Robin Tillett risked eye and wrist strain to produce the HUMAN2 and HUMAN3 data sets seen in chapter 5. Both Tony and John helped me out with field measurements, and John's skills as an impromptu milliner saved me from sun-stroke on more than one occasion.

Without the help of my family, my time at university would have been a great deal harder. So, Mom, Dad and Rachel, thanks for eight years of 'phone calls and visits, and most of all for supporting me in everything I've done for longer than I can remember.

In chapter 2, figure 2.5 shows reflectance spectra for brassica plants that is part of the

LOPEX plant spectra data set. The LOPEX data set was established during an experiment conducted by the TDP Unit of the Space Applications Institute/Joint Research Centre of the European Commission (Ispra). The `qhull` program from the Geometry Centre at the University of Minnesota was used in the production of the MRROC curves in chapter 8.

This work was supported by the BBSRC via a Silsoe Research Institute studentship.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

In 1731, Jethro Tull published his book *The New Horse Hoeing Husbandry* that contained, amongst other innovations, the designs for a device that would lead to the mechanisation of agriculture. This device was the seed drill, that allowed more orderly and reliable planting of crop than the traditional broadcasting technique. In broadcasting, the seeds are scattered randomly across the surface of the soil. Seed drilling deposits the seed on the earth in a regular row and presses it into the soil where it is more likely to be nourished and is protected from wind, rain and wildlife. The regular spacing of the rows ensures that the crop plants have approximately equal volumes of soil from which they can extract nutrients and sufficient access to sunlight, so each plant grows at a similar pace to its neighbours, and all will be ready for harvest at the same time.

In addition to its agronomic benefits, the orderly arrangement of crop ensured by the seed drill paved the way for large scale use of agricultural machines. When seeds were sown by broadcasting, the resulting irregular positioning of the crop plants meant that weed control had to be performed by manual hoeing of the field around the plants. Crops planted using the seed drill grew in rows, which meant that a hoe could be drawn by a horse in straight lines between the crop rows. Horses were gradually replaced by tractors, and for many crops, mechanical weeding using a hoe has been replaced by chemical treatment for weed and disease control. Currently, approximately £460 million are spent annually on 23,000 tonnes of agro-chemicals in the UK alone. A rise in consumer interest in organic farming, together with recent controversy over food crops that have been genetically modified to resist herbicides used in weed control, has led to pressure to reduce the amount of agro-chemical used in our farms.

Precision agriculture is a concept developed in response to these pressures, where new technology is exploited to achieve the most efficient use of agro-chemicals as possible. Instead of taking an approach where all of a field is treated with a fertiliser, fungicide or herbicide, the precision agriculture approach would be to apply differing amounts of chemical to different parts of

the field, according to local requirements. For instance, if a particular region of field was known to be vulnerable to weed infestation, then more herbicide would be directed toward that region than to other parts of the same field. The logical extreme of the precision agriculture approach is the treatment of individual plants. This approach is known as *plant scale husbandry* (henceforth PSH), and its aim is to direct treatment precisely, with no waste of fertiliser on weeds or soil, or herbicide on crop [THM96, HMT97b]. The technique may even pave the way to larger scale organic farming, with precisely guided mechanical weed control.

For PSH to be economically feasible, the treatment systems will almost certainly require autonomy from direct human supervision. Hague *et al* [HMT97a] estimate that for precise spray treatment of individual plants, a vehicle carrying the treatment system would be limited to a maximum speed of approximately $1ms^{-1}$. This limit is imposed by both real-time computational constraints and physical constraints relating to the time taken to activate a spray treatment system, and for the spray to reach the plant. Employing a farm worker to drive a vehicle at such a low speed would be prohibitively expensive, which has prompted scientists at the Silsoe Research Institute[1] to develop an autonomous horticultural vehicle as a test-bed for experiments in PSH.

This thesis reports the development and testing of computer vision algorithms designed to aid both navigation and treatment scheduling for the Silsoe vehicle. The remainder of this introduction reviews related work in mobile robotics and provides more detail on the Silsoe vehicle application, before we outline the contributions presented in this thesis and preview the contents of the next eight chapters.

1.1 Autonomous vehicles

One of the earliest applications in computer vision was that of autonomous vehicle navigation in the SHAKEY project which ran at the Stanford Research Institute from 1966 – 1972 [Nil69]. SHAKEY was a mobile robot equipped with bump sensors, a laser triangulation range-finder and a camera, and was used as a platform for artificial intelligence research into topics such as object recognition, navigation and path planning in a well-structured indoor environment. Since this early work, a great deal of research has been performed with autonomous robot vehicles, and we review some of it below. The work has been partitioned into three categories: indoor robots, outdoor vehicles and agricultural vehicles. In each case, we concentrate on solutions which require little or no modification to the system's working environment.

[1] Silsoe is the UK's national centre for agricultural engineering research.

1.1.1 Indoor robots

Much of the research conducted into mobile robot navigation has concentrated on indoor applications. The typical scenario is a well-structured indoor space such as an office [SGH+97], museum [DFBT99], or factory [BDWH+90]. In these situations, the challenges are localisation, obstacle avoidance and path planning. In our horticultural application, only localisation is of interest at the moment. Path planning is straightforward as the vehicle must simply follow the rows of crop and execute turns at the end of each row to progress to the next. Obstacle *avoidance* is not of concern because any evasive action taken in the field is likely to cause damage to the valuable crop, but obstacle *detection* will be important in any commercial system to bring the vehicle to a safe halt and prevent accidental damage to the horticultural vehicle (or the obstruction), and then to alert the farmer that there is something unexpected in the field.

Localisation, the ability to determine one's position relative to a co-ordinate frame, is vital for the success of PSH, where the horticultural vehicle must navigate safely in the field and target plant matter for treatment. Many successful mobile robot localisation systems combine both proprioceptive, or dead-reckoning, information with external views of the world. The combination of the internal and external sensing modalities helps compensate for the weaknesses in each. The readings from dead-reckoning sensors such as wheel odometry, accelerometers and gyroscopes are prone to drift, and position estimates from such measurements alone tend to become increasingly biased away from the true value as the sensor drift increases with time. External sensors such as sonar, infra-red laser ranging devices and computer vision are prone to outliers and missing readings, so estimates based upon these alone may be unstable. The combination of the two sensing modalities corrects for drift in the dead reckoning and allows inappropriate external sensor readings to be filtered out of the position estimate.

Many successful robots have been constructed using this mixture of sensors. Brady *et al* [BDWH+90] describe an incarnation of the Oxford AGV project, designed for moving pallets around factory floors. In their test system odometry is married with a ring of sonar ranging devices. The sonar sensors are used to detect range to walls and corners of a room, and these are matched, where appropriate, to an *a priori* map of the factory environment [LDW91a]. Subsequently the methods were transferred to another vehicle[HGB97], where a scanning infra-red laser was incorporated into the estimation system to provide bearing information to a number of bar-code targets placed in the robot's environment from which the robot's position can be triangulated[2]. An overview of the AGV project is given in the book edited by Cameron and

[2]In the original GEC/Caterpillar "turtle" robots, localisation was attained by using the laser scanner to obtain bearing information to bar-code targets. Much of the Oxford work concentrated on fusion of this information with

Probert [CP94].

Crowley [Cro89] describes a sonar-based robot localisation algorithm that bears some similarity to that of Leonard and Durrant-Whyte [LDW91a], although the features they extract from the sonar scans differ. Crowley fits line segments to the range readings, whilst Leonard and Durrant-Whyte search for regions of constant depth (RCDs). RCDs occur "naturally" in sonar scans, and are typically caused by concave corners in the environment.

The projects mentioned above use pre-determined maps of the environment and the robots localise themselves with respect to these maps. A more difficult problem is simultaneous map-building and localisation in unknown environments, which has been attacked using sonar by Leonard and Durrant-Whyte [LDW91b] and Rencken [Ren93], and with computer vision by Faugeras and co-workers [AF89, AF88, DF90], Harris [Har92a] and, lately, Davison [Dav98].

All of these approaches [HGB97, LDW91a, Har92a, AF89, Dav98], be they with *a priori* maps or not, have two common traits. The first is the use of what Leonard and Durrant-Whyte call *geometric beacons* as reference points in the world. These beacons may be artificial targets such as the bar-codes read by the laser scanner [HGB97], or distinctive features in the sensed data, such as RCDs in the sonar scans [LDW91a], corner features in images [Har92a], 3D points, lines or planes [AF89] or image regions detected by an interest operator [Dav98]. In our horticultural application, the regular pattern formed by the crop in the ground is a natural and reliable beacon that may be used as an aid for localisation.

The second common feature of the work presented above is the use of the Kalman filter algorithm to estimate vehicle position and, in the map building applications, beacon position. The Kalman filter is a recursive estimation algorithm based upon a predict-correct cycle. A model of the vehicle's kinematics is used to *predict* the vehicle's position after motion has occurred, and a combination of dead-reckoning and external measurements are used to *correct* the predicted position estimate. The Kalman filter algorithm is used in the Silsoe vehicle as outlined below, and will be treated in greater depth in chapter 3.

1.1.2 Outdoor vehicles

As we have seen, there are valuable lessons to be learnt from indoor navigation schemes, but there are several additional challenges for autonomous vehicles that operate outdoors. Inside buildings, the floor is smooth enough that smooth planar motion with no pitch or roll and little vibration may be assumed, artificial lighting is reasonably constant, and man-made objects typically provide strong features that may be extracted relatively easily from images or sonar scans. Outside, especially off-road, the ground plane is less reliably flat, there are fewer man-made fea-

local sensing modalities such as odometry and sonar [CP94].

tures, so features extracted from images may be less persistent throughout an image sequence, and there is no control over natural lighting.

Outdoor vehicles may be separated from those that operate in man-made surroundings, such as Durrant-Whyte's automated port container vehicle [DW96] and the many autonomous road vehicles developed in Germany [Dic98], the USA [TJP97] and Japan [Tsu94], and those vehicles that operate off-road, such as the AutoNav system [BLD+98], or Carnegie-Mellon University's various off-road vehicles [LRH94] and the NASA Mars microrover [MS97]. Agricultural vehicles are discussed separately in a section below.

The automated highways system (AHS) project has been the focus of much research in the United States [TJP97, BPT98, MM97]. The AHS project aims to reduce road accidents and increase traffic flows by placing vehicles under partial or total computer control. The work at Carnegie-Mellon has largely concentrated on problems such as road following and lane-changing for both automatic vehicle control and driver assistance [TJP97, BPT98]. Such an emphasis on lateral position estimation and control is typical of much research conducted into autonomous road vehicles [THKS88, TMGM88, DM96], although McLauchlan and Malik [MM97] describe a stereo vision system for longitudinal control. In their application, the vehicle is part of a "platoon", a line of cars travelling in close proximity at high speed, and computer control is required to prevent collision in a situation where human reaction times are too long. The lateral position sensing in the platoon application is performed by electro magnetic "pick-ups" that sense the position of magnetic nails driven into the road surface at regular intervals. A full, recent review of road transport automation world-wide is provided by Masaki [Mas98].

Off-road vehicles have a less straightforward sensing task. Road vehicles may take advantage of lane markings and road surface modifications (such as magnetic nails) to aid steering and navigation, whereas off-road autonomous vehicles must determine safe routes for travel in a much less constrained environment. Dead-reckoning is much more difficult off-road, where odometry is particularly prone to inaccuracies caused by wheel-slip, and vibration of the vehicle as it traverses rough terrain adds noise to inertial sensor readings.

Once again, Carnegie-Mellon University are leaders in this field, and have developed a number of off-road vehicles, amongst them the Navlab II vehicle [LRH94] equipped with odometry and inertial sensors for position estimation, and a laser range finger for obstacle detection. Stentz and Hebert describe goal-driven navigation [SH95] where the vehicle is provided with a map showing its start position and a goal to reach. The vehicle autonomously identifies steerable routes, marking cells in the map as untraversable, high-cost or traversable depending on whether a cell contains an obstacle, is near an obstacle or free of obstacles, and finds its way to the goal.

It has been demonstrated at speeds of approximately $2ms^{-1}$ over distances of $1.4km$.

A second off-road vehicle developed at CMU is the Nomad planetary rover, also equipped with odometry, inertial sensing and a laser range-finder [MSAW99]. In addition to these sensors it has a differential global positioning system (D-GPS) capability which enabled it to successfully explore large regions of the Antarctic on an autonomously derived route passing through a number of human demanded way-points. Carrier-phase D-GPS uses a base station in conjunction with the GLONASS satellite array to derive position estimates, and can be accurate within a few millimetres, but is very expensive and can suffer from problems with satellite occlusion in built-up areas, and if communication with the base station is lost, then the system fails. Nebot and Durrant-Whyte also demonstrate an outdoor autonomous vehicle equipped with a D-GPS system [NDW99].

Perhaps surprisingly, for our horticultural application we can learn more from the road vehicles than those designed for off-road use. The problem of following the rows of crop is closer to road lane-following than less constrained off-road navigation. For treatment purposes, however, we require precise forward distance estimates, so will be more concerned with longitudinal direction estimation and control than the autonomous road vehicles research community have tended to be.

1.1.3 Agricultural Vehicles

The demands of precision agriculture have lead to a number of prototype autonomous agricultural vehicles. Yoshida *et al* [YOI88] describe a vision system for a wheat harvesting vehicle that can follow the border between the stubble left by crop that has been harvested and the uncut crop. They also detail a system for guidance of a robot in a paddy field, where the vehicle is guided along the rows of plants by laser and detects the end of each row by locating a target with an ultrasonic sensor. No details of the accuracy of location estimates or control are given for either of the two systems.

Another automated harvesting machine, the Demeter automated forage harvester [OS97] has been developed in the US. Again, they use the line between cut and uncut crop for vehicle guidance, and are able to use vision to detect the end of the crop rows. No quantitative results for localisation precision are given, but the harvester has been demonstrated by cutting over 60 acres of alfalfa hay at speeds of 4.5 kilometres per hour (human operators typically drive harvesters between 3 and 6 kph). The authors intend to integrate a D-GPS receiver into the system to increase the reliability of the system for cases where the vision algorithms may fail.

Billingsley and Schoenfisch [BS97] demonstrate an automatic guidance system that uses the crop rows as a cue for navigation (this will be seen below to be the approach taken for the

Silsoe vehicle). They detect plant matter in colour perspective images of the field collected from a camera mounted on their tractor, and use regression techniques to fit lines to the crop rows, and determine the vanishing point of the parallel rows in the image. This information is then used to provide steering information for the vehicle. They claim that their system is able to steer with an accuracy of $\pm 2cm$ with respect to the crop rows, although these figures are obtained from a test where the vehicle followed a line of white tape along a dark floor.

The agricultural systems outlined above concentrate on guidance for steering control, estimating offset and bearing angle to a line, be that line the crop rows [BS97] or the edge of harvested crop [OS97]. This is a vital component of an autonomous plant scale husbandry system if it is to navigate along the field without damaging the crop, but we also require accurate longitudinal position estimation to allow precise direction of treatment to individual plants. Differential global positing system receivers are becoming popular in precision agriculture systems [Sta96] for directing spatially variable field operations[3], and they can sense position with an accuracy of a few millimetres [MF96]. McLellan and Friesen [MF96] claim that accuracy within the region of $2cm$ will be required for some agricultural operations. However, D-GPS with such high accuracy is still very expensive and has additional problems such as the relatively infrequent availability of readings (1 Hz), and can be prone to problems such as satellite dropout, or loss of communication with the local base station. It was decided not to use D-GPS for navigation of the Silsoe vehicle, and that longitudinal position would be estimated by dead-reckoning.

1.2 The Silsoe autonomous vehicle project

The Silsoe autonomous horticultural vehicle, pictured in a field of cauliflowers in figure 1.1, has been under development since late 1993 as a platform for the investigation of precise crop and weed treatment[TMH96]. The vehicle platform started life as a commercially produced manually controlled lightweight agricultural tool carrier, and has been fitted with computer hardware and sensors (a camera, wheel odometers and accelerometers) to allow autonomous operation in the field, and a rudimentary treatment system, the spray bar labelled in figure 1.1, for PSH experiments. Computing power has been provided by two systems. From 1993 until 1998, processing was performed by a network of transputers hosted by a laptop PC. In the Winter of 1998/1999, this rather out-moded hardware was replaced with a single processor Intel Pentium-II system running the Linux operating system, although some transputer hardware has been retained to perform data collection from the dead-reckoning sensors.

[3] An example of this is patch spraying. Regions of field (typically 5 × 5 metres square) are mapped by satellite for features such as weed density, and this information is used to control variable rates of herbicide application to the large patches of field.

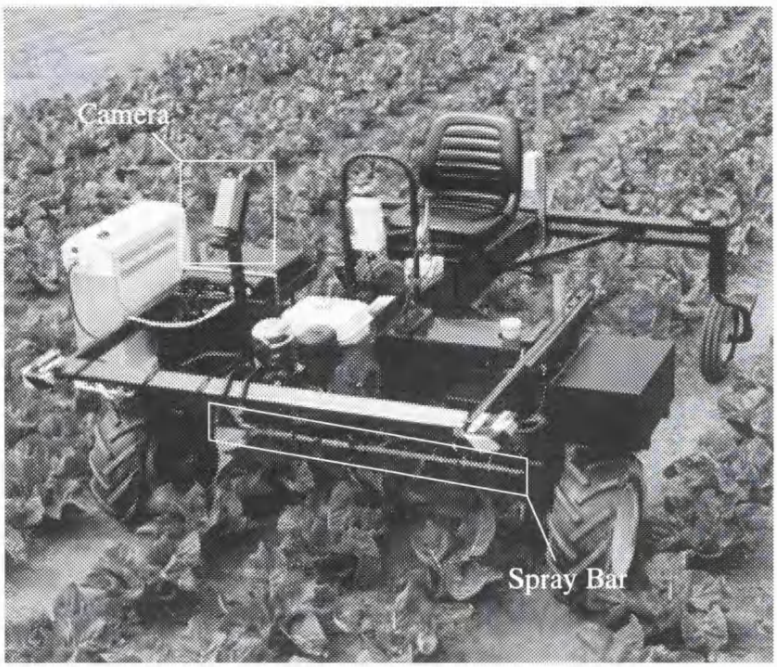


Figure 1.1: The Silsoe autonomous horticultural vehicle.

The new work presented in this thesis, started in 1996, extends the capability of the vehicle's computer vision system, and has built on a considerable volume of previous effort invested in development of the vehicle [HT96, MB95, Mar96, PSM97, BM96, SPMB96, RWBM96, HMT97a]. The remainder of this section gives details of the autonomous vehicle and identifies the problems that the work presented later in this thesis aim to solve.

For autonomous operation, the Silsoe vehicle requires two basic capabilities. The first is localisation, and the second is the ability to discriminate between the crop, weed and soil. Localisation with respect to some co-ordinate system is required if the vehicle is to navigate itself in a field of crop, and discrimination is necessary for the differential treatment of crop, weed and soil demanded by PSH. We shall treat the two capabilities separately below.

1.2.1 Localisation

Hague and Tillett [HT96] presented the vehicle control and navigation architecture, an amended and simplified version of which is illustrated in figure 1.2. Many authors are concerned with estimating vehicle position relative to an absolute cartesian frame of reference [LDW91a, SSDW95, BDWH+90], but Hague and Tillet [HT96] argue that a more natural co-ordinate system for operation of the Silsoe autonomous vehicle is one that defines position relative to the rows of crop whose treatment is the focus of the PSH. The position estimator in figure 1.2, which is a Kalman filter algorithm, calculates the distance that the vehicle has travelled along the row, the lateral

position of the vehicle with respect to the central row of crop plants and the bearing angle of the vehicle with respect to the central row.

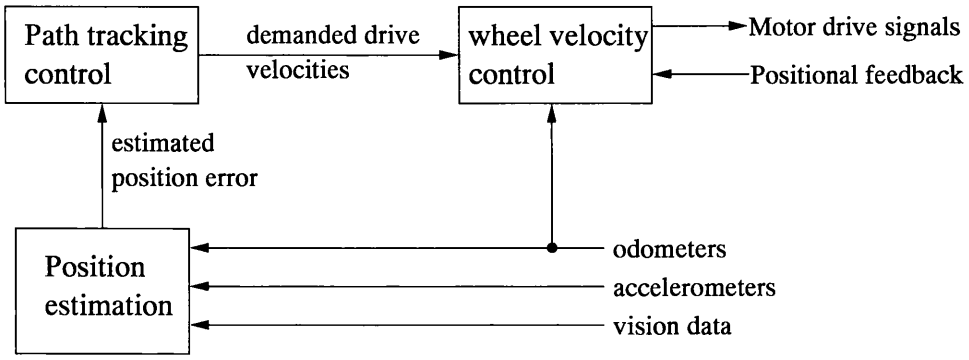


Figure 1.2: Vehicle control and navigation architecture.

Odometers and accelerometers are used to generate dead-reckoning estimates of the vehicle's position, velocity and acceleration, with periodic drift correction to the lateral offset and bearing estimates provided by the computer vision system. The vision system designed by Marchant and Brivot [MB95], described in detail in appendix A, uses the Hough transform technique [Pra91] to fit a template of the crop row structure to the image, which yields the vehicle's lateral offset and bearing relative to the rows. An input image and the fitted row model are pictured in figure 1.3. A small number of navigation trials performed by Hague *et al* [HMT97a] and Hague and Tillett [HT96], both in a test-bed environment and in a field of real crop, show that error in the estimate of lateral offset is of the order of $8 - 20\ mm$. Forward position estimates have not been assessed.

The localisation system described above is used only when the vehicle is navigating along the crop row. At the end of a bed of crop, where a bed is three parallel rows as pictured in figure 1.3, the vehicle must turn and start following the neighbouring bed. The end of each bed is detected semi-automatically; the vehicle is given the approximate length of the bed and uses its estimate of forward motion, and the number of plant matter features detected by image processing to judge when the vehicle has reached the end of the bed. The vehicle then executes a pre-programmed turn under dead-reckoning control so that it is in approximately the correct position to travel along the next bed, the vision system starts up again and visually aided tracking re-commences [HMT97a].

1.2.2 Plant recognition

In addition to localisation, a vehicle that performs PSH requires the capability to differentiate between crop plants and weed, so that treatment may be directed appropriately. To this end, Brivot

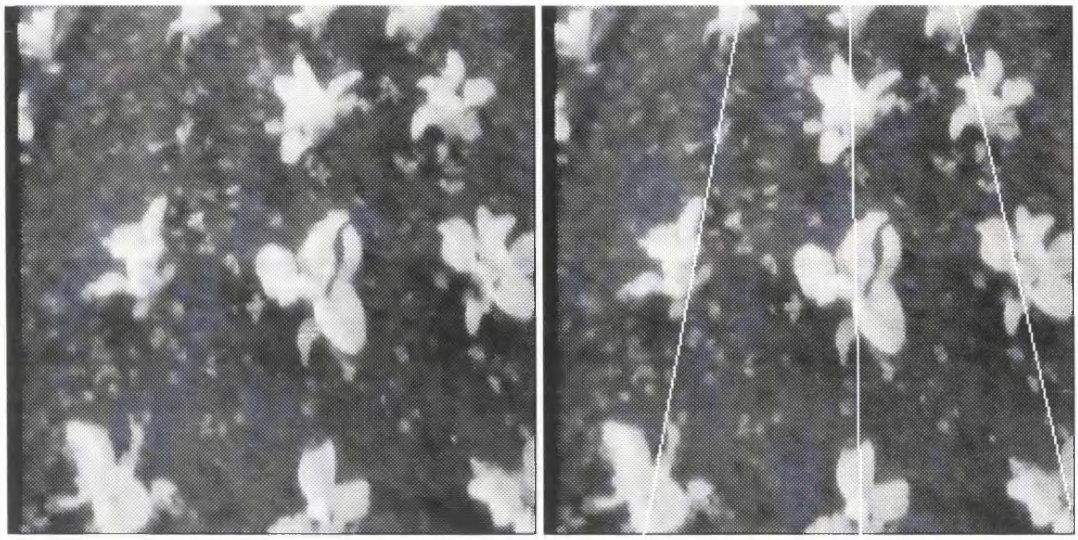


Figure 1.3: The crop row structure. Left: a view of the field from the vehicle's camera. Right: the crop rows as located by the Hough transform algorithm due to Marchant and Brivot [MB95].

and Marchant [BM96] propose candidate algorithms to separate crop, weed and soil pixels in monochrome images such as that seen in figure 1.3. The algorithms use a mixture of hysteresis thresholding of the pixel grey-levels, techniques from mathematical morphology and grey-level gradient information. Although they showed promise in off-line trials performed on a desk-top workstation with image sequences captured from the vehicle's camera, the algorithms have proved unstable in on-line operation, and are sensitive to changes in illumination and also to the size of the crop plants and weeds [Mar99].

Sanchiz *et al* [SPMB96] and Reynard *et al* [RWBM96] present algorithms for tracking individual crop plants as the vehicle traverses the field. Both algorithms have been demonstrated off-line on sequences captured from the vehicle, but have not been implemented for on-line use.

Sanchiz *et al* model each crop plant as a cluster of "blobs" extracted from the grey-level image by thresholding, and track the clusters to determine both the motion of the vehicle and the position of the plants with respect to the vehicle. The algorithm assumes that only image features that correspond to crop are given to the clustering algorithm, and relied on the method of Brivot and Marchant [BM96] to correctly provide these features.

Reynard *et al* [RWBM96] model the crop plants using contour models [CB92] and demonstrate the robustness of their tracking algorithm to temporary occlusion, as may be caused by shadows. No method of initialising contours to track particular plants was given, so again, the crop/weed discrimination problem was not solved.

1.2.3 The aims and contributions of this thesis

The work presented in this thesis aims to extend the capability of the autonomous horticultural vehicle by introducing a vision algorithm that models the crop pattern as a *grid* of crop plants rather than a set of *rows* of plants. We aim to show that this seemingly small change leads to two principal advances:

1. Forward distance estimates may be obtained by a vision algorithm that tracks the crop grid model. In the original autonomous vehicle navigation system, estimates of forward distance are generated by dead-reckoning alone, and the row model used in the Hough transform vision algorithm does not permit estimation of forward motion. Dead-reckoning systems are prone to accumulating errors that lead to biased position estimates, and problems with wheel-slip can lead to unreliable odometer measurements in the field. It is hoped that the addition of vision estimates of forward distance will correct for dead-reckoning bias.

2. The crop grid model may be used as an aid for discriminating between crop and weed plants. For reasons we shall outline in chapter 2, discriminating between crop and weed plants with image processing techniques is a much harder problem than extracting both weed and crop features from the images. However, if we are successfully tracking the crop grid, then image features that support the estimate of grid position may be assumed to be crop plants, and the remainder weeds.

The contributions made toward the autonomous vehicle project by this thesis are centred upon the development of an algorithm that tracks the crop grid model through image sequences using an extended Kalman filter. In addition to the advantages offered by a grid model over the row model, the use of the extended Kalman filter algorithm, described in detail in chapter 3, allows a more natural integration of the vision system with the vehicle's position estimator than was achieved with Marchant and Brivot's Hough transform algorithm. Furthermore, the Hough transform algorithm produces quantised estimates of vehicle position, whereas the Kalman filter tracker provides continuously valued output.

In addition to the full development of two alternative grid tracking algorithms, a method for crop/weed discrimination is developed that clusters image features under the constraint that they support the crop grid model. The use of this algorithm leads to the segmentation of images into regions of crop, weed and soil. The classified image features may be used to guide the application of treatment.

Each algorithm is implemented for off-line testing on a desk-top workstation, and performance judged on image data captured from the autonomous vehicle prior to implementation as

part of the closed loop control of the autonomous vehicle. The off-line and on-line implementations differ for practical reasons, and these differences are discussed fully before both tracking and segmentation algorithms are demonstrated in the field with real crop.

On a more theoretical note, we discuss the implications of the control theoretic concepts of observability and controllability for systems whose state is estimated by Kalman filters. A novel test for the observability of an extended Kalman filter is derived and the concept of "corruptibility" introduced, which is analogous to controllability for systems with stochastic inputs.

1.3 Thesis outline

The development and testing of the tracking and segmentation algorithms is broken down into seven chapters. The first of these, chapter 2, introduces the crop grid model and discusses the perspective imaging of the crop through the vehicle's camera. Two thresholding algorithms are proposed for extracting plant matter features from the image data, and their performance compared across a set of test images using receiver operating characteristic analysis, which also aids the selection of an operating point for the preferred thresholding algorithm.

The recursive Kalman filter estimation algorithm, and the related extended Kalman filter, are discussed in chapter 3. We define and discuss the issues of controllability and observability and their implications for Kalman filtering. Corruptibility is defined, and a novel test for the observability of extended Kalman filters derived.

The details of the extended Kalman filters implemented in this thesis are given in chapter 4. The process and observation models for the off-line implementations are given, and the transfer to the on-line system discussed. The filters are analysed in terms of their corruptibility and observability. The off-line tracking algorithms are demonstrated in chapter 5, together with initial navigation trials of the on-line system in a simplified indoor test-bed environment.

In chapter 6, we address two practicalities that are important when using extended Kalman filters: initialisation and data association. The extended Kalman filter is a recursive algorithm where the latest estimate of the state of a process is a combination of the previous estimate and some measurements of the process of interest. The purpose of filter initialisation is to 'bootstrap' the recursion with an initial estimate of the vehicle's location relative to the crop grid that will subsequently allow successful tracking. Data association techniques are required to extract appropriate features from the image data to use in estimating the crop grid position. Feature validation is also discussed.

Whilst chapters 4 and 5 concentrated on the use of the crop grid model as a cue for navigation, chapter 7 shows how it can be used to classify image features as crop or weed. The algo-

rithm is tested on image sequences captured from the vehicle at different stages of crop growth and in differing weather conditions.

Two final experiments are presented in chapter 8, where the navigation and crop/weed discrimination algorithms are tested on a bed of real crop. Quantitative analysis of the on-line system's navigation performance and spray treatment precision is given, and a qualitative analysis of crop/weed discrimination.

We close with chapter 9, which reviews the results of our experiments and discusses their implications for plant scale husbandry. Directions for future work are identified.

Chapter 2

Image processing in the field environment

In chapter 1, we introduced the Silsoe autonomous vehicle project as a test-bed for experiments in plant scale husbandry. The vehicle is equipped with a computer vision algorithm that uses a model of the rows of crop plants [MB95], in conjunction with a dead-reckoning estimator, to provide localisation information that allows the vehicle to navigate along the rows of crop plants in the field. In this thesis, we propose that using a crop grid model allows additional localisation information to be derived from the images (in the form of forward distance estimates) and also aids crop/weed discrimination.

In this chapter, we review modelling techniques in computer vision to help us specify a model of the crop grid structure. The crop is viewed through a camera mounted on the front of the vehicle, and the perspective imaging of the scene is discussed, and formulae are presented which relate the ground plane position of the crop grid to the locations of crop plants in the image.

We also discuss the processing of the images captured by the camera. To locate the crop grid model, it is necessary to extract the crop plant features from the image. In plant scale husbandry, we aim to treat weeds as well as (but in a different manner to) crop, so it is necessary to extract all plant matter from the image. Two grey-level thresholding algorithms are proposed for this purpose, and their performance on images collected from the vehicle is analysed using receiver operating characteristic curves. This method allows both overall performance comparison and a solution to the problem of operating point selection.

2.1 The semi-structured field environment

As noted in chapter 1, the field environment in which the autonomous vehicle operates contains a natural "beacon" to navigate by in the form of the crop grid planting pattern. For agronomic reasons the crop, which in the examples given throughout this thesis is cauliflower, is planted in a grid structure which allows each individual plant space to grow and access to a region of soil from which it draws nutrients. The standard practice is to grow seedlings in a greenhouse, and to

transplant them into the soil at a later date. This planting process is performed mechanically by a transplanter, which places the seedlings on an approximately regular grid, a perspective view of which can be seen in figure 2.1. This approximate structure, imposed on the world by the crop planting can serve as a cue for the two aims of vehicle control:

Navigation: the structure provided by the crop grid pattern may be used as a beacon to localise the vehicle relative to the crop. This localisation serves as an aid to the dead-reckoning system for vehicle navigation.

Treatment: the knowledge that crop plants are arranged in this grid structure may be used to discriminate between crop and weeds. Crop plants should lie within the grid structure where they were planted, whilst weeds will usually grow elsewhere, both because they tend to grow at random and because they will not thrive if germinating close to a well established seedling.

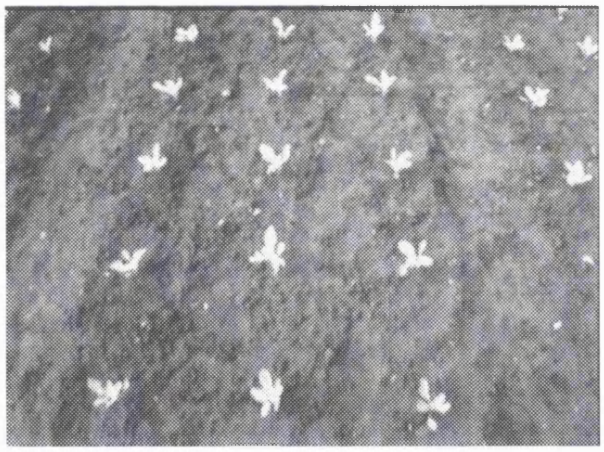


Figure 2.1: The grid planting pattern as captured from the vehicle camera.

The image in figure 2.1 has been taken from a sequence captured from the camera mounted on the vehicle. By using models of the grid structure and the motion of the vehicle, this grid structure will be tracked as the vehicle traverses the field. The following sections discuss object modelling in machine vision, whilst the problem of tracking receives attention in chapter 3.

2.1.1 Model based tracking in machine vision

Model based tracking is a powerful method of reducing the computational load on an image-processing system. Rather than analysing all of a static image and categorising each object in it separately, a model-based tracker aims to 'lock on' to a specific object or objects in an image sequence and use *a priori* knowledge of the appearance, and sometimes also the dynamics, of the

object(s) to track it (them) throughout the sequence. Several different types of model have been proposed [Har92b, CTCG92, YCH89, Low90, BCZ93] with varying degrees of flexibility and rigidity, and it is important that an appropriate one is chosen, in order to capture the variability within the class of shapes modelled without introducing extraneous degrees of freedom. Subsequent sections review different modelling schemes that are popular in machine vision whilst bearing in mind that the task at hand is to model the grid structure of the crop planting pattern.

2.1.2 Active contour models

A popular technique used in many tracking applications in machine vision is based upon the active contour model (ACM). Originally proposed by Kass *et al* [KWT87], the ACM, or "snake", is a string of control points, coupled by an elastic material having tension and stiffness properties, which performs a local search in an image for features such as edges. In its original formulation, the snake is said to have converged when the "forces" caused by the internal stiffness and tension balance those derived from the edge features in the image. A drawback of the snake algorithm is that it requires a good initial configuration close to the required edge, otherwise it will simply relax to a straight line. The original tracking mechanism for the snake relied on small inter-frame motion. The predicted position of the snake in the next image of a sequence was simply the position at which the snake had converged in the current image. Later, more sophisticated predictive motion models were incorporated into the tracking mechanism [TS92, Bau96].

The idea of joining the ends of the contour to form a closed loop, or "balloon" with an added pressure force was then proposed [SWHB90, Coh91]. The pressure force makes the balloon expand until it meets a strong edge, so the initial position is no longer so critical. Also, owing to the internal smoothing forces, the balloon will ignore weak edges that may be caused by image noise. A further development of the balloon model is the statistical snake. This is an active region model (ARM) [IP94], which allows adaptive control of the pressure force to encourage the snake expand or contract in order to surround regions of an image conforming to a statistical model. This particular technique has been successfully applied to segmentation and tracking of road and track regions using statistical colour models [Ale98] and also to colour and texture segmentation [ZLY95].

The ACM incorporates a very simple shape model where the object of interest is a smoothly bounded region of the image enclosed by a perimeter of edge features. Such a model is sensitive to clutter in the image. A more robust tracker that is less sensitive to clutter could be realised if the shape of the contour were further constrained to be specific to a particular class of objects. Curwen and Blake [CB92] describe such a method, where a B-spline ACM[1] (or "dy-

[1]This is essentially a B-spline where the control point positions are influenced by the image. Owing to the implicit

namic contour") is coupled to a template such that, in the absence of image forces, the contour's shape is that of the template. This biases the contour to lock on to shapes in the image that are similar to the template. The models are permitted to deform under affine transformations, and a Kalman filter (see chapter 3) is used to track the contour position over time (similar motion models and tracking schemes have been proposed by Baumberg for more regular ACMs [Bau96]). This method was demonstrated to run in real-time on a small network of transputers. Since publication of the original work [BCZ93], the method has been developed extensively to incorporate features such as learning the dynamics of the object from training sequences [BIR94], and also differentiating between changes in object shape in the image that are caused by actual change of the object shape and apparent changes owing to object motion and change of viewpoint [RWBM96]. A further development is the CONDENSATION algorithm [IB96] used as the estimation mechanism in place of the Kalman filter. The strength of the CONDENSATION algorithm is that, unlike the Kalman filter which is restricted to the uni-modal Gaussian, it has the ability to handle multi-modal probability densities, thus enabling multiple hypotheses to be propagated through the tracking process. In particular, the capacity to maintain multiple hypotheses increases the robustness of the tracking process. For example, if the tracker becomes distracted by clutter in an image, a uni-modal tracker can fatally lose track because its single hypothesis allows only one object position to be represented. A multi-modal model, however, allows both the clutter positions and true object position to be represented, and when the clutter is found not to satisfy the dynamics of the model, then the true object mode will re-assert itself via propagation of the distribution through the state evolution model.

Although active contour models have been used for tracking individual plants [RWBM96], and despite their success in many practical vision systems, active contour models and active region models are not natural candidates for the application at hand, which is tracking the position of the structure formed by a group of plants. When used to track the outline of an object, the ACM fits its control points to features (typically edge segments) and presents an interpolated outline of the object. The ACM method is not well suited to the crop grid tracking application of interest to this thesis, because the structure to be tracked is not a region or object enclosed by a boundary, but rather a set of individual crop plant positions.

2.1.3 Point distribution models

The point distribution model (PDM) [CTCG92] represents an object by the position of a set of, say n, landmark points derived from distinguishing features on its boundary or interior. The model is trained on several images, where landmarks have been specified by hand, which aim

smoothness of the B-spline, the stiffness and tension forces of the Kass type snake are not required [BCZ93].

to represent typical variations in the object's appearance. This is usually performed by aligning the training images using a Procrustes procedure [CTCG92] and carrying out a principal components analysis on the distribution of the aligned landmark points obtained from the training images. This determines the mean shape of the object together with vectors describing its statistically independent principal models of variation. Only the most significant modes are retained in the model and those accounting for little variation are ignored in order to limit the number of model parameters, thereby increasing efficiency and avoiding over-fitting problems.

When coupled with an image feature search strategy, the PDM is known as an Active Shape Model (ASM). The feature search strategy is a method for fitting the model to a new instance of the object in an image. In fact, the original ASM paper [CT92] was subtitled "Smart Snakes", highlighting the similarities between the ASM and ACMs discussed above, as both can be regarded as a collection of landmark or control points on an object whose positions are influenced by various "forces". In an ACM, the control points are connected by elastic forces and there is no shape specificity. In the ASM formulation, the landmark point positions are coupled by the statistics underlying the PDM, which permits only certain modes of shape deformation. Since it allows the object shape to vary and since the variability may be learnt from example training data, active shape models have been used in the tracking of many types of biological objects, such as hands [Hea95], pigs [TOM97], fish [MT97] and flocks of ducks [SBT97]. With its ability to cope with deformations, the PDM/ASM is especially well suited to modelling the changes of shape caused by object flexibility, but it has also found application in fitting a generic model of an object to particular types. For example, Ferryman *et al* [FWSB95] construct a generic car model with basic features such as bonnet and boot of variable size and shape, and then train it on several different models of car, and use the resulting PDM in a system to lock on to and track any type of car present in the scene.

Another option would be to use the PDM for the crop grid tracking problem with landmark points taken as the plant positions, but some care must be taken with the model construction. The Procrustes alignment procedure described by Cootes *et al* [CTCG92] uses a similarity ransform (planar translation and rotation) to align the training shapes before the principal components analysis is performed. In our application, where we image the crop in perspective, a different transformation is required. Chatterjee and Buxton [CB98] use an affine transformation in the Procrustes alignment of examples of the outlines of drivable regions that have been extracted from images of unsurfaced country lanes. Although such an approach has not been taken in this thesis, it might be possible to use the perspective projection in an alignment scheme prior to computation of the mean of the aligned crop grid examples and computing their modes of

variation in the usual manner. Chatterjee and Buxton also make the observation that the affine transformations between road surface and camera are not entirely random, because they are determined, by driver action, to keep the vehicle on the road. To capture this systematic variation they also performed PCA on the transformations [CB98].

2.1.4 Rigid models

The planting pattern is an example of man-made *structure* imposed on the world. The tracking of man-made *objects* has been the focus of much attention [Ste89, Low90, Har92b]. Such objects are often rigid or have simple mechanical internal degrees of freedom and may usually be represented by CAD-like models. Stephens [Ste89] used a Hough transform method to track an object grasped by a robot arm. A full 3D model of the object was constructed, together with a look-up table to indicate which features (points on the object's edges) should be seen from particular viewpoints. Although the pose estimates generated by the algorithm were noisy, the system was quite successful in dealing with occlusions and cluttered backgrounds. It should be noted that by its nature the resolution of any Hough transform pose estimator is limited by the quantisation of the Hough accumulator, and also that Stephens' system had no means of capturing the dynamics of the model. The assumption was that there was little inter-frame motion and that the edge positions could be found in successive frames by local search from their previous locations. Lowe [Low90] extracted edge segments from the whole image and a non-linear minimisation procedure was used to fit a 3D model onto the image. In this case, both rigid objects and articulated objects with rigid subcomponents were modelled, although no quantitative analysis of the results was given.

The RAPiD algorithm [Har92b] uses similar models to those of Stephens [Ste89] (points on object edges, with view graphs to predict occlusions for certain poses) to track aircraft and other objects. The tracking method employed here is the Kalman filter, and the algorithm produces high quality pose estimates in image sequences of both real and model objects. The Kalman filter yields continuous (i.e. unquantised) pose estimates, and also a measure of uncertainty on the estimate.

Both the RAPiD algorithm and Stephens' Hough transform method are designed for rigid objects of ideal shape. The crop grid pattern may be regarded as a rigid structure, because once the crop is planted it is fixed in position. However, the feature points (i.e. the crop plants) should be allowed to occupy non-ideal positions owing to the variability in the planting process. Any model of the grid structure should allow for this uncertainty on individual plant positions, which leads us toward the flexible template model.

2.1.5 Flexible templates

The flexible template [YCH89] is a relatively simple idea. A parameterised model of the object of interest is constructed, and the parameters are varied so as to fit the model to the image. Yuille *et al* [YCH89] introduced the deformable template and applied it to extracting features such as the eyes and mouth from images of human faces. Figure 2.2 illustrates their template for the eye. All of the labelled parameters are allowed to vary in a scheme which minimises a potential energy defined from the goodness of fit of the model to the image. The model of the crop

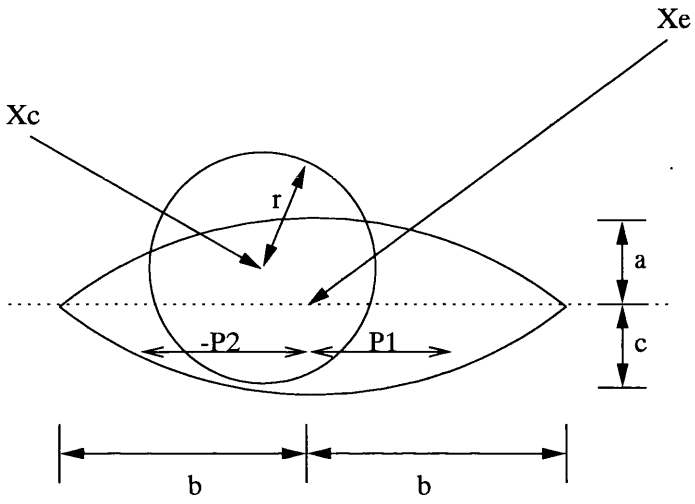


Figure 2.2: A flexible template model of the human eye (adapted from Yuille *et al* [YCH89]).

grid structure presented below bears similarities to both a rigid model and a flexible template, depending on whether or not its parameters are allowed to vary.

2.2 Modelling and viewing the grid structure

The model of the crop grid pattern is depicted in figure 2.3, where the black circles represent the ground plane positions of the set of cauliflowers currently in the field of view of the camera. The figure also shows two sets of axes; (x_w, y_w) is the world co-ordinate frame with the world z axis, z_w, projecting into the page, and (x_c, y_c, z_c) are axes which belong to a co-ordinate system attached to the camera mounted on the vehicle, with z_c describing the camera's optic axis. Figure 2.4 shows an alternative view that clarifies the relationship between the two co-ordinate frames. The crop is assumed to lie on the ground plane $z_w = 0$, and the y_w axis runs through the central row of crop in the field of view. Errors arising from the assumption that the crop lies in the ground plane are discussed in section 2.3.8 below.

It should be noted that the world co-ordinate frames are local to the vehicle co-ordinate system, rather than being fixed to a particular origin in the world. Operations such as weed control

and crop treatment are carried out by the vehicle, so it is sensible to describe the locations of the crop and weed in a co-ordinate system relative to the treatment system. As the vehicle moves across the field, the world axis x_w axis moves in direct proportion to the distance travelled along the crop rows. A set of co-ordinates is marked on the diagram which specifies the position of

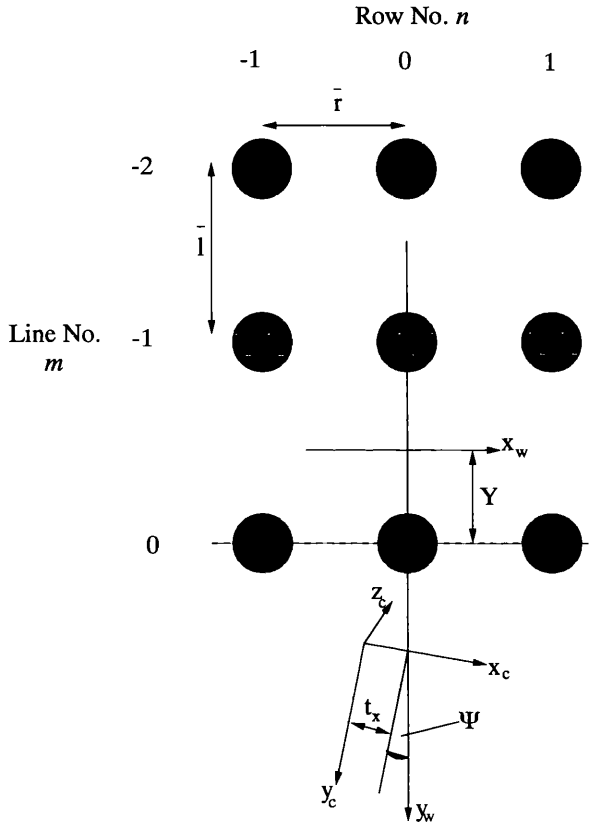


Figure 2.3: The grid model.

the vehicle relative to the crop. These co-ordinates are t_x, the offset of the optic axis from the central row of crop, Ψ, the vehicle's bearing relative to the central row of crop, and Y, which is the offset between the world x_w axis and the bottom-most crop plant in the current image. Y is the co-ordinate which anchors the grid to the vehicle's co-ordinate system. Two grid parameters, the mean inter-row spacing $\bar{r}$ and inter-line spacing $\bar{l}$ (figure 2.3) are used in conjunction with indices m and n to generate the world co-ordinates of the individual plants, as shown in equations 2.1 and 2.2.

$$x_w = n \times \bar{r}, \tag{2.1}$$

$$y_w = m \times \bar{l} + Y. \tag{2.2}$$

Marchant and Brivot [MB95] (appendix A) presented equation 2.1 in their work on using a Hough transform to track the row structure. As noted in chapter 1, their algorithm modelled the planting pattern as a set of rows only. In this thesis, the use of a grid model facilitates the

identification of individual plants within the rows, as well as enabling the vision system to be used to estimate the forward position of the vehicle via Y. Estimates of the vehicle's forward motion were not possible in the Hough transform approach.

In chapter 4, two tracking algorithms are developed. The first estimates only the relative position of the crop to the vehicle, i.e. (t_x, Y and Ψ). In this case, the grid model is closely related to the rigid models of Harris [Har92b]. In the second tracker, the grid parameters $\bar{r}$ and $\bar{l}$ are also estimated. In this case, the model is closer to the deformable template [YCH89], although the fitting of $\bar{r}$ and $\bar{l}$ is performed by an extended Kalman filter instead of via the minimisation of a potential energy function.

2.2.1 Perspective imaging of the grid structure

The crop grid structure is imaged by a monochrome CCD camera mounted on the front of the vehicle, with its optic axis pointing in the direction of motion at an angle ϕ to the normal to the ground plane, as illustrated in figure 2.4. The images of the crop formed in this manner clearly show perspective effects as seen in figure 2.1 above.

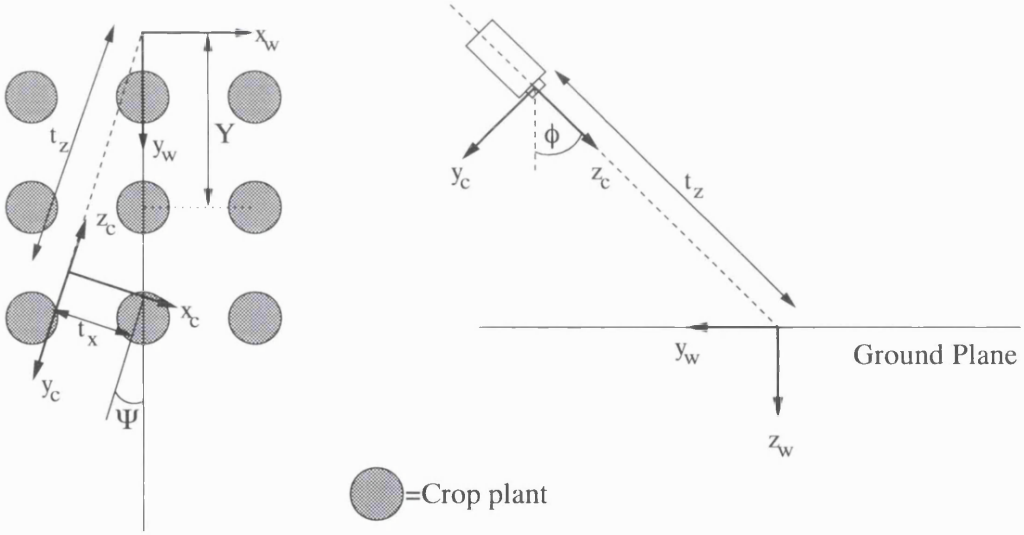


Figure 2.4: The camera and world co-ordinate systems.

To model the imaging geometry, we begin with the perspective imaging equations of Tsai [Tsa86] and, following Marchant and Brivot [MB95] (appendix A), arrive at the following expressions for the 2D image plane co-ordinates (x_u, y_u) of ground plane features $(x_w, y_w, 0)$:

$$x_u = f\frac{(x_w \cos\Psi + y_w \sin\Psi + t_x)}{x_w \sin\Psi \sin\phi - y_w \cos\Psi \sin\phi + t_z}, \tag{2.3}$$

$$y_u = f\frac{(-x_w \sin\Psi \cos\phi + y_w \cos\Psi \cos\phi + t_y)}{x_w \sin\Psi \sin\phi - y_w \cos\Psi \sin\phi + t_z}. \tag{2.4}$$

Equations 2.3 and 2.4 were derived by Marchant and Brivot [MB95] from those given by Tsai, with the added assumption that the camera does not pitch or roll (i.e that the camera axes (x_c, y_c, z_c) of figure 2.3 do not rotate about the world axes x_w or y_w). This assumption is made because the vehicle is running on well tended tilled fields, and it is confirmed from observation of several long image sequences. In equations 2.3 and 2.4, ϕ is the angle between the camera's optic axis (z_c) and the world z_w axis, whilst Ψ is that shown in figures 2.3 and 2.4. f is the focal length of the camera's lens. The quantity t_z is distance along the optic axis between the camera's optical centre and the point at which the optic axis intersects the ground plane. t_y gives the offset, in camera co-ordinates, of the point where the optical axis intersects the world x_w axis and, as illustrated for our system in figure 2.4, may be set to zero. One further assumption is that the angle Ψ is small enough for the approximations $\cos\Psi \approx 1$ and $\sin\Psi \approx \Psi$ to hold. This is typically the case when the vehicle is moving along the rows of crop.

From the above, it is possible to generate image pixel co-ordinates (x_f, y_f) for each plant centre (m, n) by combining equations 2.1 and 2.2 with 2.3 and 2.4, and inserting a suitable estimate of the vehicle position (t_x, Y, Ψ) and crop grid parameters $\bar{r}$ and $\bar{l}$:

$$x_f(t_x, Y, \Psi, \bar{r}, \bar{l}, m, n) = \frac{f}{dx} \frac{n\bar{r} + \Psi(m\bar{l} + Y) + t_x}{n\bar{r}\Psi \sin\phi - (m\bar{l} + Y)\sin\phi + t_z} + C_x, \tag{2.5}$$

$$y_f(t_x, Y, \Psi, \bar{r}, \bar{l}, m, n) = \frac{f}{dy} \frac{(m\bar{l} + Y - \Psi n\bar{r})\cos\phi}{n\bar{r}\Psi \sin\phi - (m\bar{l} + Y)\sin\phi + t_z} + C_y. \tag{2.6}$$

The values of dx and dy in equations 2.5 and 2.6 give, respectively, the horizontal and vertical dimensions of the camera pixels, whilst (C_x, C_y) is the co-ordinate (in pixels) of the centre of the imaging surface. It should be noted that equations 2.5 and 2.6 do not correct for lens distortion. Tsai's original model and camera calibration algorithm do allow for radial lens distortions, but previous work from the autonomous vehicle project has suggested it is not necessary to include such factors here [Mar96].

2.3 Image processing

Given the crop grid structure and imaging geometry, the expected image position of crop plants can be generated using equations 2.5 and 2.6 as described in the previous section. Image processing is then used to provide plant features to which the crop grid model may be fitted.

Ideally, we would be able to extract the crop plants and weeds separately, and fit the crop grid model to the crop plant features alone. However, even plants of the same species and age may have different numbers of leaves or different sized leaves which can overlap in different ways, so there is a great deal of variability within a species. It would be difficult to build a

model that could capture such variability without over-generalising to other species. An option might be to model an individual crop plant leaf and locate all of the leaves in the image. Nielsen *et al* [Nie94] and Abbasi *et al* [AMK97] both present schemes for recognising leaf outlines, either using point distribution models [Nie94] or a curvature scale-space representation [AMK97]. However, both of these systems rely on the leaves being imaged one at a time when they are laid flat. In the field, the leaves are three-dimensional objects that will most likely be partially occluded, and neither system addresses these issues.

An active region model [IP94] (section 2.1.2 above) driven by a colour or texture model would be a useful solution for extracting one species of plant, provided that a suitable colour or texture model could be found. Zwiggelaar [Zwi98] conducted a survey of research using spectral techniques for plant species recognition with a view to their potential use in crop/weed discrimination. The conclusions of this review were that red and near infra-red wavelengths were most effective for species identification, but that a classifier based on spectral information alone was unlikely to prove robust. As we shall see below, spectral information is useful for differentiating between plant matter and soil, but it would appear that it is not the solution for crop/weed discrimination.

Unfortunately texture modelling is not viable either. As visual inspection of figure 2.7 will confirm, our current imaging system does not have sufficient resolution to allow textural differences between crop and weed plants to be discerned. However, Soille [Soi99] has used morphological operators on higher resolution non-perspective colour images to extract cauliflowers with some success. The algorithm uses the distinctive vein structure found in the cauliflower leaves as a cue for segmentation, but this is not visible in the near infra-red perspective images captured from the vehicle.

One cue that we can use, however, is scale. The weeds are typically smaller than the crop plants, and we will exploit this property in the crop/weed discrimination algorithm described in detail in chapter 7. Until we are able to model the plants more effectively, we aim to extract features describing both crop and weed from the images and use the data association techniques discussed in chapter 6 to select the features used for fitting the crop grid model.

Our image processing algorithm for plant feature extraction comprises two stages. The first stage aims to extract crop and weed plant matter pixels from the image – this is performed by a grey-level threshold algorithm described below. The second stage is feature extraction. Pixels classified as plant matter are grouped into "blobs" in the image (a blob being a set of connected neighbouring pixels), and the centroid and size of these blobs are used to characterise them as crop plant or weed. The crop/weed classification process is detailed in chapter 7.

Sections 2.3.1 – 2.3.6 present the thresholding algorithm and methods for performance evaluation and parameter setting of this algorithm. Feature extraction, and potential problems with the feature extraction process are then discussed (sections 2.3.7 and 2.3.8).

2.3.1 Thresholding IR images

Biller [Bil98] has noted that contrast between soil and plant matter in monochrome images captured in the near infra red spectrum is greater than contrast in monochrome images taken in the visible spectrum. The reason for this is illustrated in figure 2.5, which shows the reflectance spectrum for a plant of the brassica family (of which cauliflower is a member) together with the reflectance of soil. The brassica data comes from the LOPEX data set [HJA+95], whilst the soil spectrum is modelled as a straight line using data from Biller [Bil98]. No reflectance data is available for the weed population, but the brassica data has characteristics typical of most plant species. In the visible spectrum (c. 400 – 700 nm), the reflectance of plant matter and soil are quite similar, soil being more reflective on the whole than the crop. However, in the near infra-red section of the spectrum (particularly in the 760 – 850 nm region), the reflectance of the plant matter is much higher than that of the soil[2]. This will lead to strong contrast between plant matter and soil in well illuminated images captured at these wavelengths (a typical monochrome CCD camera is sensitive to wavelengths of up to 1100 nm). For this reason, a filter is placed over the vehicle's camera which blocks visible light and passes infra-red alone. The enhanced contrast makes the use of a grey-level threshold a viable plant matter/soil discrimination technique on the near infra red images thus formed. We now aim to demonstrate the effectiveness of thresholding on example images captured in the field at different stages of plant growth and under different weather conditions by using receiver operating characteristic curves to analyse the results of thresholding.

2.3.2 Evaluation of the segmentation algorithm

To assess the utility of grey-level thresholding for classification of these images into plant matter and soil regions, and also to aid the choice of a suitable threshold level, receiver operator characteristic (ROC) curves [GS66, vT68] are used. The discriminatory power of a classification algorithm controlled by a single parameter is reflected by the area under the ROC curve, and the operating point for the controlling parameter may be selected using the slope of the ROC curve. The use of the area under an ROC curve is described below, whilst parameter selection is discussed further in section 2.3.6.

The reflectance characteristics shown in figure 2.5 would indicate that plant matter should

[2]Unfortunately, consistent separation of crop and weed plant species using spectral reflectance is not achievable [Zwi98].

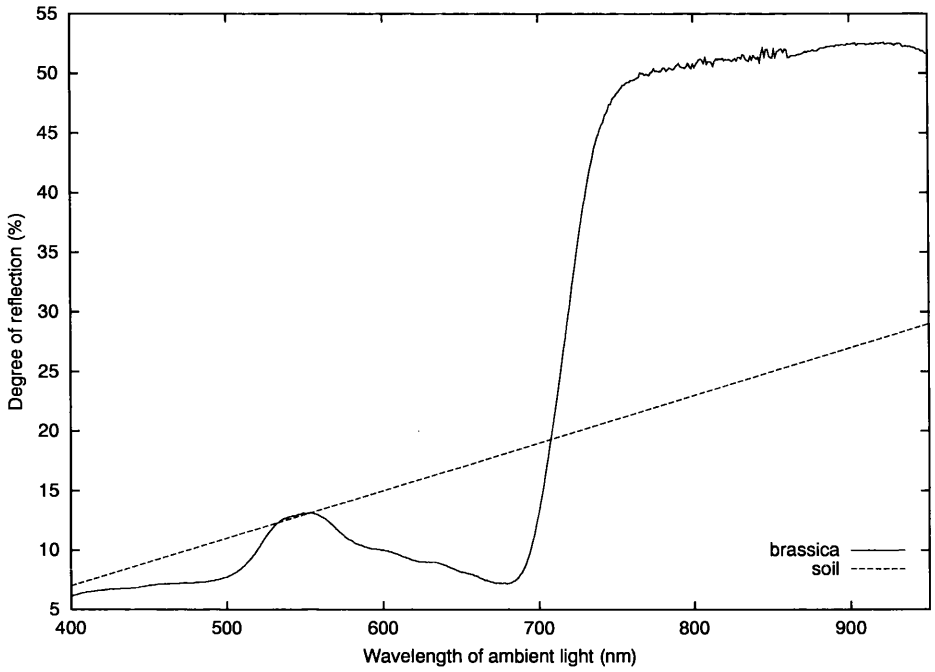


Figure 2.5: Reflectance of plant matter and soil. Solid line – reflectance of brassica plant material. Dashed line – soil reflectance.

be considerably brighter than the soil under the infra-red illumination present in natural sunlight. In the image, the grey-levels of the plant matter pixels should thus be higher than those of the soil pixels. If this is so, then grey-level should be a useful discriminant between the two populations. A rank order statistic, known as the Wilcoxon statistic (sometimes known as the Mann-Whitney statistic) [vdW69] tests the null hypothesis that a variable is *not* useful for discriminating between samples from two populations. If this hypothesis is not proven, then the variable is a useful discriminant. The Wilcoxon statistic is described here because it has been shown to be equivalent to the area underneath an ROC curve [HM82]. The equivalence arises from the fact that both quantities measure the probability that in randomly paired soil and plant matter pixels, the grey-levels will allow the two pixels to be correctly identified.

In subsequent sections, image pixels are separated into two classes; plant matter composed of both crop and weed, and soil. In the language of two class discrimination problems, the plant matter pixels are considered to be "positives" and the soil pixels to be "negatives". Both classes may be characterised by a parameter g (in our case the pixel grey-level). If we denote the parameter value of a negative case by g_N and the parameter value of a positive case by g_P, over the population of positives and negatives it is traditionally stated that

$$g_P > g_N, \tag{2.7}$$

i.e. that g_P is "stochastically larger" than g_N. This is assumed in the following discussion. In

the Mann-Whitney test, the null hypothesis is that the values g_P are *not* statistically higher than the values g_N, i.e. that

$$Pr(g_P > g_N) = 0.5, \tag{2.8}$$

where $Pr(\cdot)$ denotes probability. This is the probability that is estimated by the Wilcoxon statistic. If g is a suitable discriminant, then this probability will be much closer to one than to 0.5[3].

In the case of discrete data (such as pixel grey-levels), Hanley and McNeil [HM82] showed that the Wilcoxon statistic may be computed from a set of comparison scores. Compare each pair of pixels g_P and g_N, and score the comparison as follows

$$S(g_P, g_N) = \begin{cases} 1 & \text{if } g_P > g_N \\ \frac{1}{2} & \text{if } g_P = g_N \\ 0 & \text{if } g_P < g_N \end{cases} . \tag{2.9}$$

To compute the Wilcoxon statistic for a set of n_P positive examples and n_N negatives, the following formula is used:

$$W = \frac{1}{n_P \cdot n_N} \sum_{1}^{n_P} \sum_{1}^{n_N} S(g_P, g_N). \tag{2.10}$$

For two populations that are completely separable by parameter g, this statistic will have the value 1; if they are completely overlapping, then the null hypothesis that g is not a useful discriminant is true, and the value of W is 0.5. The closer W lies to 1, the better the two data sets may be separated using the parameter g as a discriminant.

Hanley and McNeil [HM82] demonstrate the equivalence between the Wilcoxon statistic and the area under the ROC curve. Thus, the area under the ROC curve will indicate whether the parameter g is suitable for distinguishing between two data sets. Given the area under an ROC curve θ, it is stated that

$$\theta = W = Pr(g_P > g_N). \tag{2.11}$$

The formation of the ROC curve for the binary classification of images is now discussed.

If a pixel classification algorithm uses a single threshold τ to control its performance, then an ROC curve is produced by varying this threshold or decision parameter and analysing the performance of the algorithm at each operating point. To analyse the performance, a set of test images is required for which there is a ground truth classification, with every pixel labelled either positive or negative[4]. These test images are used for comparison with automatic classification results. A number of settings of the threshold τ are chosen and the algorithm applied to a test image. At each setting, it is possible to place every pixel into one of four groups:

[3] If the probability is close to zero, then the two data sets are distinct, but $g_P < g_N$.

[4] In practice, it may be wise to exclude certain pixels from the ground truth classification; this is discussed further in section 2.3.4.

1. True Positives (TP): plant matter pixels correctly classified as plant matter.

2. False Positives (FP): soil pixels incorrectly classified as plant matter.

3. True Negatives (TN): soil pixels correctly classified as soil.

4. False Negatives (FN): plant matter pixels incorrectly classified as soil.

From these four classes, two figures can be derived; the true positive ratio

$$\text{TPR} = \frac{\text{number of TP}}{\text{number of TP + number of FN}}, \tag{2.12}$$

and the false positive ratio

$$\text{FPR} = \frac{\text{number of FP}}{\text{number of FP + number of TN}}. \tag{2.13}$$

The TPR gives the probability that a positive (plant pixel) will be *correctly* classified at a threshold setting, whilst the FPR is the probability that a negative (soil pixel) will be *incorrectly* classified at the same setting. Because they are probabilities, TPR and FPR both lie in the range $(0, 1)$.

For each setting of the threshold τ, a (TPR,FPR) pair is generated which are plotted against each other to form the receiver operator characteristic (ROC) curve. We can similarly produce an ROC curve for an entire image test set by calculating TPR and FPR on the basis of classified pixels from each image in the set. Owing to the range of TPR and FPR, the curve is bounded by the unit square (the ROC space), and it is also always possible to find parameter settings that generate the extreme points (0,0) and (1,1), where every pixel is classified as negative or positive respectively. The area under the ROC curve is denoted θ and, as stated in equation 2.11, is equivalent to the Wilcoxon statistic [HM82].

Whilst the Wilcoxon statistic W tests the separability of two data sets using a parameter g, the area θ underneath an ROC curve generated over a range of algorithm parameter settings τ reflects the ability of the algorithm to separate the data. An algorithm with the worst possible classification performance produces an ROC curve with area $\theta = 0.5$. Such an algorithm randomly classifies each pixel as positive or negative with equal probability regardless of the true class of the pixel. When θ = 1.0, the algorithm is perfect, and the two sets are classified without error. ROC curves for these two extreme cases are illustrated in figure 2.6. The ROC curve for the perfect classifier runs along the TPR axis from (0,0) to (0,1) and then between the points (0,1) to (1,1), whilst the line running along the (0,0) – (1,1) diagonal is the ROC curve for the classifier which does no better than random chance. For this reason, the (0,0) – (1,1) line is often known as the "chance diagonal".

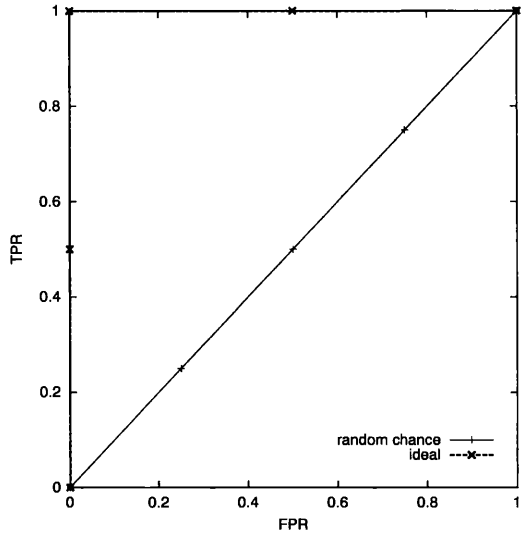


Figure 2.6: The ideal and random chance ROC curves.

If we have two (or more) rival algorithms designed to perform the same segmentation task, then the area θ is a suitable metric to aid the selection of one of these algorithms. The algorithm which produces the ROC curve that encloses the greatest area will provide the best overall classification performance for a range of threshold or decision parameter settings.

2.3.3 Image sequences and ground truth

Throughout this thesis, four sequences of images captured from the vehicle are used for off-line testing of classification algorithms. An example image from each sequence is given in figure 2.7 (a)–(d). The sequences have been chosen to represent a range of crop growth stages and imaging conditions, although this range should by no means be considered exhaustive. Sequence A consists of 960 images of crop that is approximately 8 weeks old, with few weeds, recorded on an overcast day, sequence B has 960 images of 3 week old plants in overcast conditions with very few weeds present, sequence C contains 1380 images of 6 week old crop with a moderate weed density, and sequence D (1280 images) is again of 3 week old crop with few weeds, but this time in bright sunlight. The resulting shadows can be seen clearly in D. To test the effectiveness of thresholding these images, a subset of each image sequence was chosen such that no two images are of overlapping areas, which ensures that no two pixels in the test set represent the same patch of soil or plant. For each image in these test sets, a ground truth labelling was produced by hand segmenting the image pixels into three classes: crop, weed and soil. The ground truth images can then be used for comparison with the output of the automatic classification algorithms. The crop and weed ground truth images are combined to provide a plant matter labelling or "mask" image, used in the evaluation of the thresholding algorithms presented below. An example image

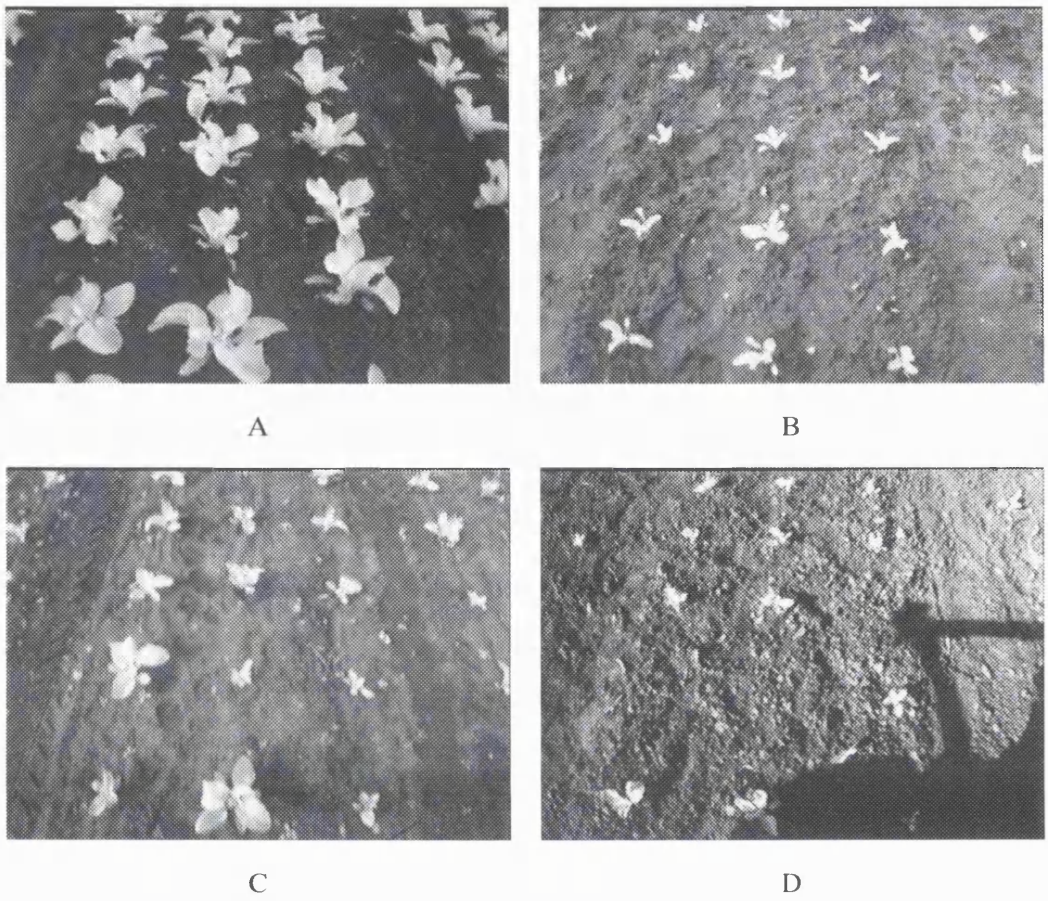


Figure 2.7: Examples from the four image sequences A – D.

from sequence C and its plant matter mask can be seen in figure 2.8, where the soil is black and plant matter is white. As can be seen from the mask image in figure 2.8, only the plants in a trapezoidal area of the image have been segmented by hand. This is because the segmentation algorithm described in chapter 7 operates only on those features that lie within the area between the wheels of the vehicle. This is the region that undergoes treatment from the vehicle's spray bar, and it is determined by using knowledge of the vehicle's dimensions and position relative to the crop.

2.3.4 Segmentation experiments – algorithms and measurements

Two algorithms have been tested for the segmentation of plant matter from soil on the basis of infra-red image grey-level value. The first algorithm is a straightforward threshold on the grey-

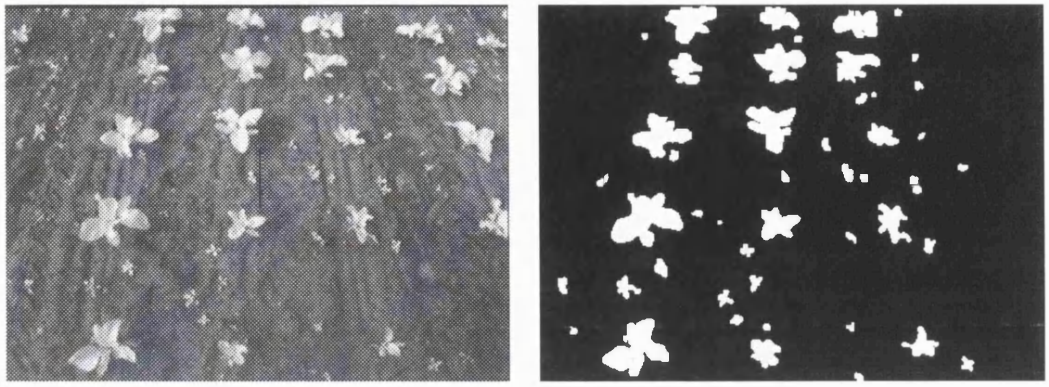


Figure 2.8: An image and its plant matter mask.

level, where the value of the output pixel $O(x_f, y_f)$ is given by

$$O(x_f, y_f) = \begin{cases} 255 & \text{if} \quad I(x_f, y_f) \geq \tau \\ 0 & \text{if} \quad I(x_f, y_f) < \tau \end{cases}, \tag{2.14}$$

where $I(x_f, y_f)$ is the corresponding pixel from the input image and τ is the threshold setting that is varied to produce points on the ROC curve. According to equation 2.14, bright pixels above or equal to the threshold τ are labelled white, as plant matter, whilst the duller pixels are labelled black, as soil.

The second algorithm, proposed by Hague [Hag98], is based on the observation that there appears to be a brightness gradient across the ground plane in many of the example image sequences. The cause of such a gradient is most likely the position of the Sun relative to the ground plane and the vehicle's camera, and the interaction of the illuminant with the rough surface of the soil. Accurate modelling of illumination and reflectance effects is a complex issue and not of concern in this thesis. More complex models are known for surface reflectance, such as those due to van Branniken *et al* [vBSK98] or Oren and Nayar [ON95].

The thresholding algorithm proposed by Hague uses a linear approximation of this gradient as a function of the y_f (i.e. vertical pixel co-ordinate) position of each pixel in the image. The algorithm is also adaptive to the average brightness of the image, which will offer some robustness to changes in illumination as, for example, when the Sun is temporarily masked by a cloud. A mean grey-level is computed for both the top (μ_1) and bottom (μ_2) halves of the image and these two means are used as fixed points to linearly interpolate a mean $\mu(y_f)$ across the height of the image. The value of the output pixel $O(x_f, y_f)$ is now given by the adaptive interpolating

thresholding algorithm:

$$O(x_f, y_f) = \begin{cases} 255 & \text{if} \quad I(x_f, y_f) \geq \alpha\mu(y_f) \\ 0 & \text{if} \quad I(x_f, y_f) < \alpha\mu(y_f) \end{cases}, \tag{2.15}$$

where α is a gain that is used to set the threshold level. α is varied to produce points on the ROC curve. As will be demonstrated by the results below, this simple algorithm offers an improvement on the straightforward fixed threshold of equation 2.14.

Before we present the results and comparison of the two thresholding algorithms, a comment on the ground truth images used is required. The ground truth images are produced by hand using standard image-editing software, and are subject to error, especially at border pixels where plant matter is adjacent to soil. Some of these pixels will be soil misclassified by hand as plant matter, some plant matter pixels will be misclassified as soil, and some pixels will genuinely be of mixed class. Alexander [Ale98] noted such problems with border pixels and proposed that at the border between foreground (in our case plant matter) and background (soil), regions of doubt should be inserted, and the pixels within these doubt regions should be ignored for the purpose of assessing classifiers. This strategy is followed here. All pixels that are on the border of plant matter and soil in the ground truth images are assigned to the doubt class and ignored in the classification assessments.

To demonstrate the effect of including and excluding the doubt regions on classification performance, figure 2.9 shows three ROC curves for the performance of the threshold described by equation 2.14 when applied to image sequence C. The solid line plots the ROC curve that results when the border pixels are excluded from the classification, the dashed line shows the ROC curve obtained when the border pixels are included as soil, and the dotted line plots the ROC curve when border pixels are included as plant matter. The area under the curve is greatest when the doubt pixels are excluded from classification: 0.9846, compared with 0.9790 when border pixels are classified as soil, or 0.9842 when they are classified as crop matter. Since this effect is caused by uncertainty in the ground truth classification, such border pixels should be ignored when assessing algorithm performance.

2.3.5 Segmentation experiments – results

Figure 2.10 shows the ROC curves generated for the fixed threshold algorithm (dashed), and the adaptive interpolating threshold algorithm (solid) on each of the four data sets. Each curve was generated using 27 parameter settings; 27 settings of the threshold τ (equation 2.14) for the fixed threshold algorithm, 27 settings of the gain α for the adaptive interpolating threshold algorithm (equation 2.15).

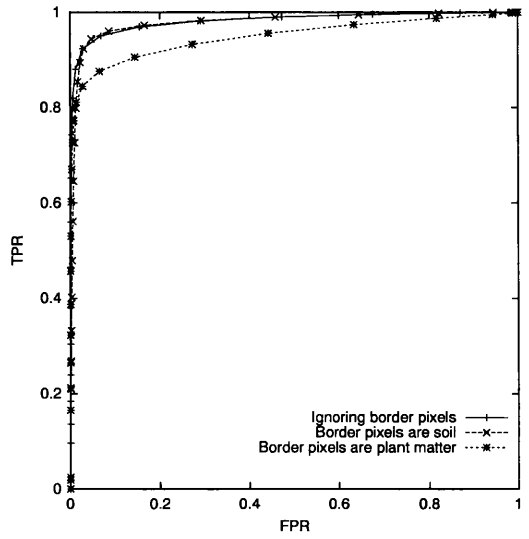


Figure 2.9: The effect of border pixels on classifier performance.

The area under each ROC curve calculated using the trapezium rule is given in table 2.1. Alexander [Ale98] has demonstrated empirically that the area underneath an ROC curve calculated by this method is insensitive to the set of thresholds (or gains) chosen to generate the curve. Different choices of thresholds were found to lead to variation in area only in the 6^{th} decimal place. A more principled analysis of the standard error of the area under the ROC curve is presented by Hanley and McNeil [HM82]. However, for this calculation to be computationally tractable on data sets of the size under consideration here (millions of pixels), knowledge of the underlying probability distributions of the positive and negative classes is required. For this reason, the areas are presented to the 4^{th} decimal place, reflecting the level of confidence used by Alexander [Ale98].

Data Set	Fixed	Adaptive
A	0.9958	0.9957
B	0.9749	0.9779
C	0.9816	0.9846
D	0.8835	0.9241

Table 2.1: Area underneath ROC curves for the fixed threshold and adaptive interpolating threshold algorithms for data sets A–D.

Inspection of the figures in table 2.1 and of the curves in figure 2.10 shows that for sequences A–C the performances of the fixed threshold algorithm and the adaptive interpolating threshold algorithm are very similar. The adaptive threshold algorithm performs marginally bet-

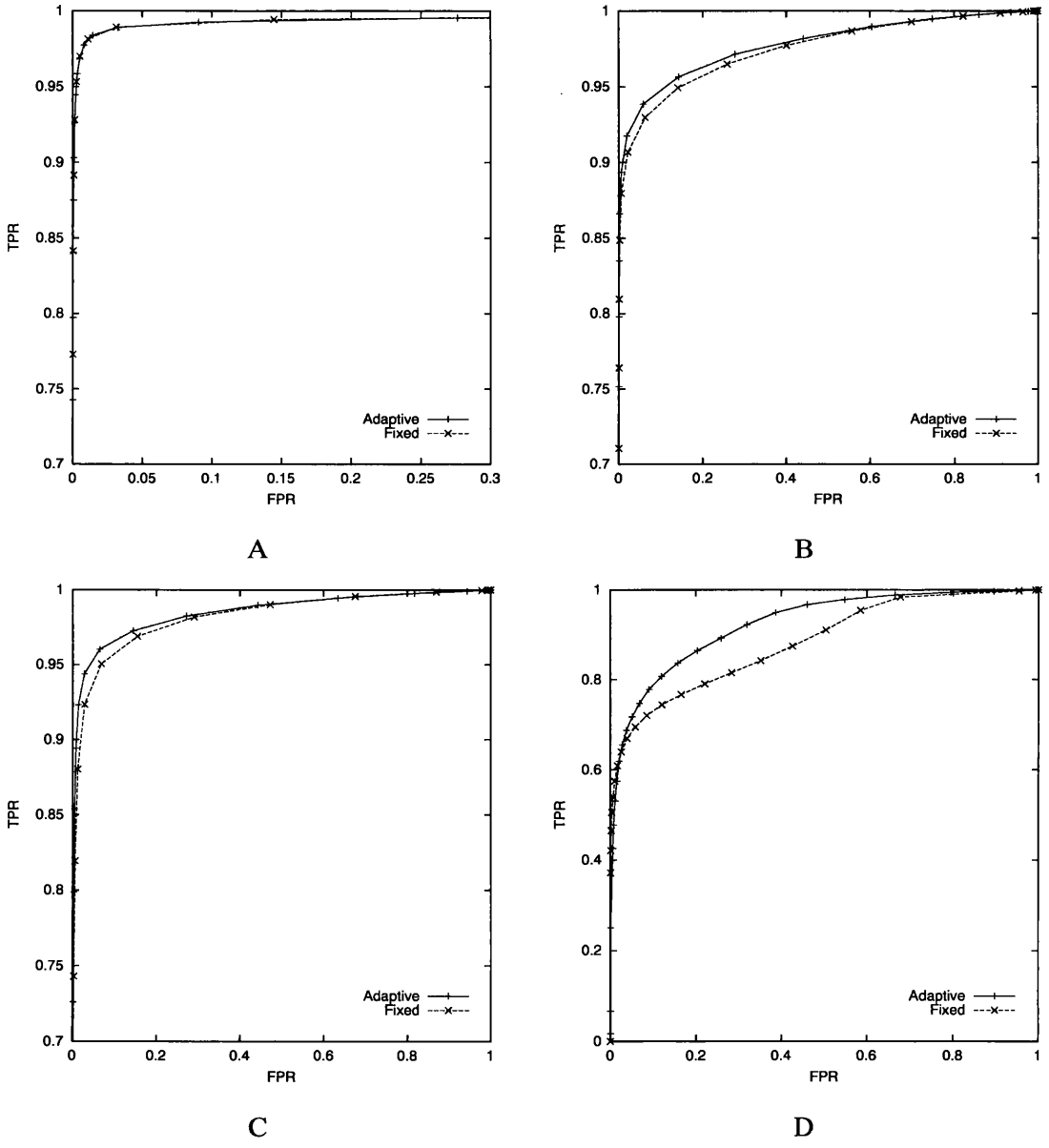


Figure 2.10: Experimental results, data sets A–D. Solid line – adaptive interpolating threshold. Dashed line – fixed threshold. Note the changes of scale between plots.

ter than the fixed threshold algorithm for sequences B and C, whilst the fixed threshold is very slightly superior for sequence A. The adaptive interpolating threshold is noticeably better with an area of 0.9241 *vs.* 0.8835 for the fixed threshold for sequence D. Interpolation across the height of the image would appear to compensate for the shadows present in sequence D (figure 2.7 D shows an example), even though the shadow only extends across the bottom quarter of the image. For this reason, the adaptive interpolating threshold algorithm is selected for image processing in preference to the fixed threshold.

2.3.6 Threshold gain selection

The area under an ROC curve offers a measure of the effectiveness of an algorithm over a range of operating points. The slope of the ROC curve may be used to select the operating point of the algorithm (in our case, the threshold gain α). The "best" operating point is the one which minimises the Bayes risk attached to a decision. Bayes risk combines the probabilities of correct and incorrect decisions together with the costs incurred by making an error and the values associated with proper classification. van Trees [vT68] gives the following formula for the Bayes risk associated with a decision strategy,

$$\mathcal{R} = \mathrm{C_{FP}P_N}\Pr(\mathrm{FP}) - \mathrm{V_{TN}P_N}\Pr(\mathrm{TN}) + \mathrm{C_{FN}P_P}\Pr(\mathrm{FN}) - \mathrm{V_{TP}P_P}\Pr(\mathrm{TP}), \tag{2.16}$$

where $\mathrm{C_{result}}$ is the cost of a particular incorrect decision, $\mathrm{V_{result}}$ the value of particular correct decision. P_{class} is the prior probability of a positive (P) or negative (N) case occurring ($\mathrm{P_P} + \mathrm{P_N} = 1$), and $\Pr(\mathrm{result})$ is the probability that the outcome of the thresholding decision will be FN, FP, TN or TP. In fact, these probabilities are provided by

$$\Pr(\mathrm{TN}) = 1 - \mathrm{FPR}, \tag{2.17}$$

$$\Pr(\mathrm{FP}) = \mathrm{FPR}, \tag{2.18}$$

$$\Pr(\mathrm{TP}) = \mathrm{TPR}, \tag{2.19}$$

and

$$\Pr(\mathrm{FN}) = 1 - \mathrm{TPR}, \tag{2.20}$$

where TPR and FPR are given by equations 2.12 and 2.13. If we now substitute the four expressions above into equation 2.16, and differentiate it with respect to the threshold gain α, the following expression is obtained

$$\frac{\partial \mathcal{R}}{\partial \alpha} = \mathrm{P_N}(\mathrm{C_{FP}} + \mathrm{V_{TN}})\frac{\partial \mathrm{FPR}}{\partial \alpha} - \mathrm{P_P}(\mathrm{C_{FN}} + \mathrm{V_{TP}})\frac{\partial \mathrm{TPR}}{\partial \alpha}. \tag{2.21}$$

Equation 2.21 describes how the Bayes risk $\mathcal{R}$ changes with variation of the threshold gain α. The risk is at a minimum when this expression is set to zero, which occurs when the slope of the ROC curve, $\partial TPR/\partial FPR$, is given by

$$\frac{\partial TPR}{\partial FPR} = \frac{\mathrm{P_N}(\mathrm{C_{FP}} + \mathrm{V_{TN}})}{\mathrm{P_P}(\mathrm{C_{FN}} + \mathrm{V_{TP}})}. \tag{2.22}$$

Thus, the best operating point of a classifier is the threshold gain at the point on the ROC curve where the slope is given by equation 2.22. This result requires that the prior probabilities $\mathrm{P_P}$ and $\mathrm{P_N}$ of the occurrence of the positive and negative classes respectively are known, and also that the values and costs of correct and incorrect decisions are available. These values and costs are application specific and will ideally be set by a domain expert.

The adaptive interpolating threshold algorithm has a gain parameter, α, that was varied to create the points on the ROC curves plotted in figure 2.10. For each sequence a value of this gain may be chosen using the gradient methods outlined in section 2.3.2, provided that prior probabilities for the occurrence of soil and plant pixels are known together with the costs and values of incorrect and correct decisions.

The prior probabilities of the different types of pixel occurring may be obtained from analysis of the ground truth images (once again, the doubt category pixels are excluded), but no such simple prescription exists for assessment of the costs and values, which must be set according to the requirements of the task. In fact, if we look back at equation 2.22, which sets the slope for optimal performance, it can be seen that the individual costs and values are not necessarily required to set the operating point, provided that the "costs ratio"

$$\Phi = \frac{\mathrm{C_{FP}} + \mathrm{V_{TN}}}{\mathrm{C_{FN}} + \mathrm{V_{TP}}} = \frac{\partial \mathrm{TPR}}{\partial \mathrm{FPR}} \frac{\mathrm{P_P}}{\mathrm{P_N}}, \tag{2.23}$$

is known. If the costs ratio is greater than 1, then the emphasis is on the correct classification of negatives; if the costs ratio is less than one the emphasis is on the correct classification on positives.

In our case, navigation requires that the image processing provides features that support the crop grid model of figure 2.3. It should extract image features that accurately represent the crop plants with few false positive (i.e. misclassified soil pixels), especially at the edges of the plants. Equation 2.23 can be used to set the value of Φ. The parameter α was tuned by hand on images from sequence C[5], and the slope of the ROC curve for the adaptive interpolating threshold algorithm on sequence C was found (several values of α around the tuned point were used

[5]The parameter α was set such that the plant matter in the image was well represented in the resulting binary image, whilst keeping down the number of incorrectly classified soil pixels.

to generate a dense sampling of the ROC curve, and the slope was calculated by central differences). The value of α derived as described above by hand is 1.64, corresponding to a costs ratio Φ of 3.644. The values of α set with $\Phi = 3.644$ for each sequence are given in table 2.2 below, together with the TPR and FPR associated with the given threshold gain.

Sequence	α	TPR	FPR
A	1.32	0.95623	0.00311
B	1.73	0.82434	0.00073
C	1.64	0.83421	0.00409
D	2.91	0.28627	0.00135

Table 2.2: Threshold gains α for each image sequence.

In table 2.2, where the costs ratio is set at 3.644, it can be seen that for sequences A–C, over 80% of plant matter pixels are correctly identified (TPR), with less than 0.5% soil pixels misclassified (FPR). Although a correct classification rate for plant pixels of 80% may seem low, perusal of figures 2.11 – 2.13, where an example segmented image from each image sequence is presented along side its ground truth plant matter image, shows that most of the plants are well represented in the automatically segmented image. For sequence D, only 28.6% of plant

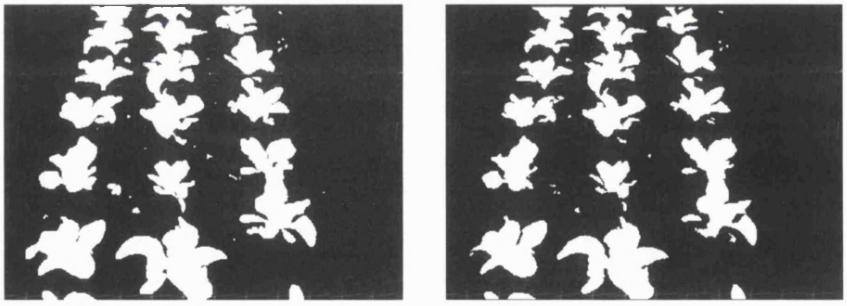


Figure 2.11: Example segmentation for sequence A. Left: ground truth. Right: automatic segmentation.

pixels are correctly classified, although the misclassification rate of soil pixels is only 0.135%. The example segmented image and its ground truth plant matter image can be seen in figure 2.14 where, as might be expected from the TPR and FPR figures in table 2.2, many of the plant matter pixels are missed, and there are also very few misclassified soil pixels.

The costs ratio was set for sequence C such that the crop plant pixels were well represented in the segmented images; in the absence of strong shadows, this costs ratio would seem suitable, as the results for sequences A and B bear out. If the performance of the segmentation algorithm

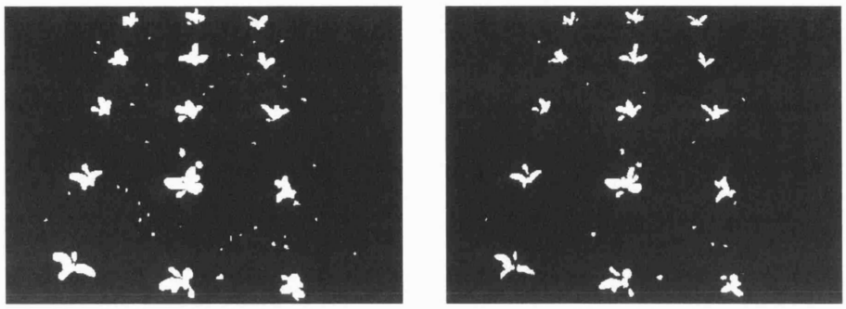


Figure 2.12: Example segmentation for sequence B. Left: ground truth. Right: automatic segmentation.

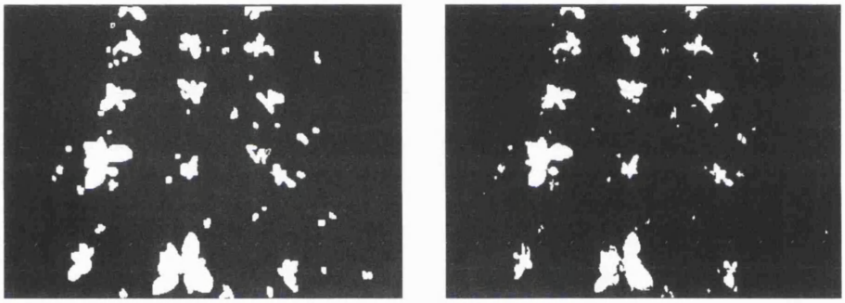


Figure 2.13: Example segmentation for sequence C. Left: ground truth. Right: automatic segmentation.

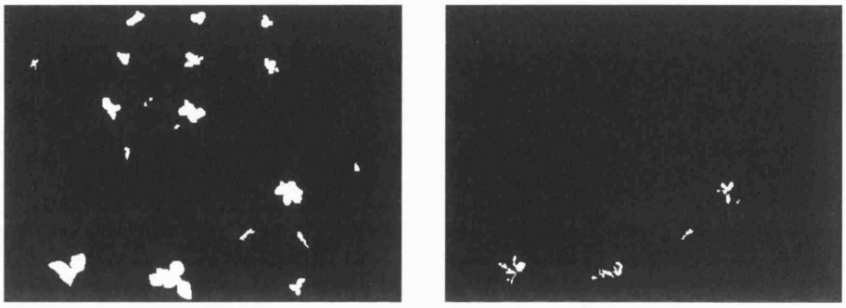


Figure 2.14: Example segmentation for sequence D. Left: ground truth. Right: automatic segmentation.

were invariant to the effects of shadows, the costs ratio would ensure similar performance on sequence D. However, the figures in table 2.2 and the example image of figure 2.14 illustrate that the algorithm performance is not invariant to shadows, and the costs ratio set on sequence C leads to many plant pixels being misclassified. By setting $\alpha = 2.15$, more plant matter pixels are correctly classified whilst keeping the number of misclassified soil pixels under 2%. The costs ratio corresponding to this operating point is 0.322; the TPR and FPR figures are given in table 2.3, and an example segmentation in figure 2.15.

Sequence	α	TPR	FPR
D	2.15	0.599396	0.017743

Table 2.3: Threshold gain, TPR and FPR for sequence D with new costs ratio.

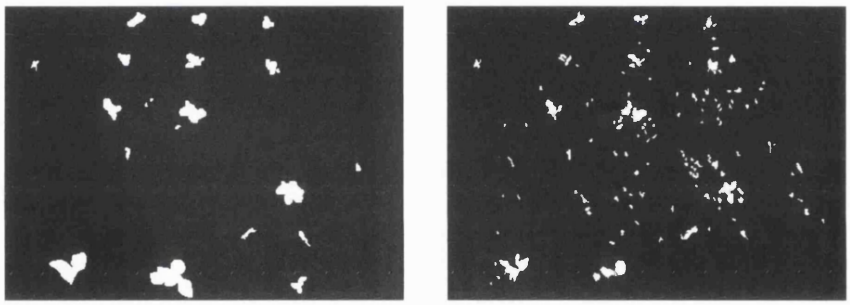


Figure 2.15: Example segmentation for sequence D with revised cost ratio. Left: manual ground truth image. Right: automatically segmented image.

Before moving on to how we extract features from the plant matter images, a comment on the cost ratios for sequences A – C and D should be made. In sequences A – C, the costs ratio is 3.644, so the emphasis is on the correct classification of soil pixels; in these evenly illuminated images the plant matter pixels are easily distinguished, so the challenge for the classifier is to reduce the number of soil pixels classified as plant matter. For sequence D, the ratio drops to 0.322, which places an emphasis on the correct classification on plant pixels. The bright sunlight present throughout sequence D results in deep shadows cast by the vehicle, and also specular highlights on the soil; the result of these effects is that it is harder to extract plant matter from the images, so the classifier is required to weight correct classification of plant matter more.

2.3.7 Feature extraction

The thresholding algorithm described above takes images as input, and produces images (albeit simple ones) as output. Figures 2.11 – 2.15 show typical thresholded images, where, ideally,

every white pixel is plant matter, and every black pixel soil. The pixels are clustered into "blobs" in the image, and each of these blobs will be interpreted as either a crop plant or a weed by the algorithms presented later in chapter 7. These algorithms classify the blobs on the basis of their position on the ground plane relative to the crop grid, and their size. The position of each blob is found by locating its centroid in the image and projecting this on to the ground plane via the camera calibration. The size of a blob is calculated by counting the number of pixels in that blob.

The feature extraction process takes the binary images, clusters the pixels into blobs and produces a list of centroids and areas describing those blobs. We have chosen to use the chain-coding algorithm due to Freeman [Fre61] to perform these actions. Alternatives such as region growing or split and merge algorithms [Pra91] would give the same results on such simple binary images.

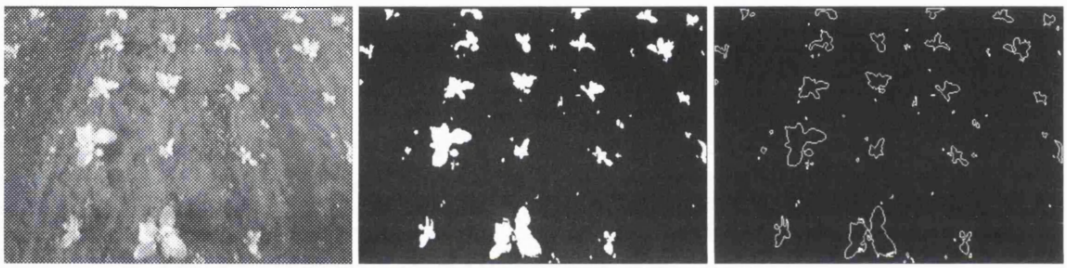


Figure 2.16: An image, its automatically extracted plant matter and the contours extracted by chain-coding the image.

The operation of the chain-coder is sketched in figure 2.16. The original image is seen on the left, with the thresholded binary image in the centre. The image on the right of figure 2.16 shows the blob contours extracted by the chain-coding algorithm. The centroid and size of each blob is easily computed from the outline.

2.3.8 Problems with feature extraction

The feature extraction techniques described above are simple and fast, and therein lies their appeal for a real-time application such as vehicle navigation. The simple nature of the algorithms mean that some types of error are likely to occur. Pixels of different classes (representing crop, weed or soil) may become clustered into a single blob – each blob is interpreted, and classified, as a single type of plant, so some of these pixels will be misclassified. Furthermore, the centroid of a blob containing a mixture of pixels will not necessarily correspond to the centre of a plant, nor will the blob's size in pixels reflect the size of a plant. The centre of the plant is taken to be the root of the plant for navigation purposes, as the root gives the grid position in which the

cauliflower was originally planted. Other errors caused by the use of a simple threshold are illustrated below, together with a note on the interpretation of the blob centroids as points in 3D space. The experimental results of chapters 5, 7 and 8 demonstrate that the algorithms developed later in this thesis for vehicle navigation and image segmentation show considerable robustness to these errors in many circumstances.

The centroid of the blob features is interpreted by the tracking algorithm as the position of the plant's centre on the ground plane. Obviously, when a blob consists of a mixture of pixels from different classes, the centroid is not the centre of an individual plant. Notably, when the crop plants grow above a certain size, they start to touch in the image, which will lead to them being merged during feature extraction. An example of this from sequence A is illustrated in figure 2.17, where four plants merge into a single outline, and are therefore represented by a single centroid and area. This aspect of the feature extraction obviously limits the operating range of the autonomous vehicle. If a sizeable proportion of plants start to touch in the image, the features extracted from the image no longer relate to the crop plant centres required by the crop grid tracker. At this stage, the vehicle will be unable to track the grid, so navigation and segmentation will not be possible. This situation is one of the more compelling reasons for finding a more sophisticated image processing technique that will allow the extraction of plant centroids directly from the grey-level image data.

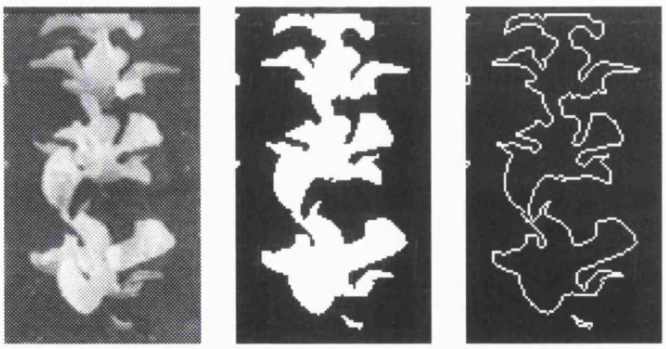


Figure 2.17: Large plants merge into a single feature. Left: image detail. Middle: thresholded detail. Right: feature outline.

The opposite of the merging of several plants can also occur. A single plant may become fractured into several disparate features. If there is a dark region (such as a shadow) between the main part of a plant and one of its leaves, then this leaf will be fractured from the plant body by the thresholding algorithm, and will generate a separate feature. An image detail from sequence C where this occurs is presented in figure 2.18. Chapter 7 contains an algorithm for clustering

split features into single plants.

Figure 2.18: A plant fractures upon thresholding. Left: image detail. Middle: thresholded detail. Right: feature outlines.

One further problem pertains to the interpretation of the blob centroids in terms of their 3D position. The centroids are translated from screen to world co-ordinates using the camera calibration, together with the assumption that they lie on the ground plane, i.e. have $z_w = 0$. In fact, the plants are complex three-dimensional objects, and the centroid of each image blob represents a point in 3D space above the ground plane. When these true 3D positions are interpreted as ground plane positions, a projection error will occur. Unless the shape of the plants is known, it is impossible to exactly quantify the magnitude of these projection errors. An estimate of the errors may be obtained, however, by modelling the crop as a simple shape, such as an inverted hemisphere (to mimic the spreading of the cauliflower's leaves), then rendering a scene of these hemispheres as if viewed through the vehicle's camera. Results from such a test with hemisphere radius $150mm$ yielded a maximum error on the y_w co-ordinate of $178.0mm$ (minimum $44.3mm$), and on x_w co-ordinate of $53.2mm$ (minimum $0mm$, along the camera's optic axis). To reduce the magnitude of these errors, we construct a "virtual ground plane" above the soil and project the feature centroids onto this, as illustrated with inverted hemispheres in figure 2.19. By setting the height of the virtual ground plane equal to that of the hemispheres, the projection errors drop to a maximum magnitude of $90.8mm$ in the y_w direction (minimum $50.0mm$) and $23.0mm$ maximum error in the x_w direction (minimum $0.0mm$, again along the optic axis). For smaller plants such as the weeds which lie closer to the ground plane than the crop, projection onto the virtual ground plane will increase the feature localisation error. However, this is acceptable for navigation because it is the crop plants whose position directly affects the estimate of crop grid location. For precise treatment of weeds, however, the use of the virtual ground plane may be undesirable, because of the enlarged projection errors for plants closer to the true ground plane.

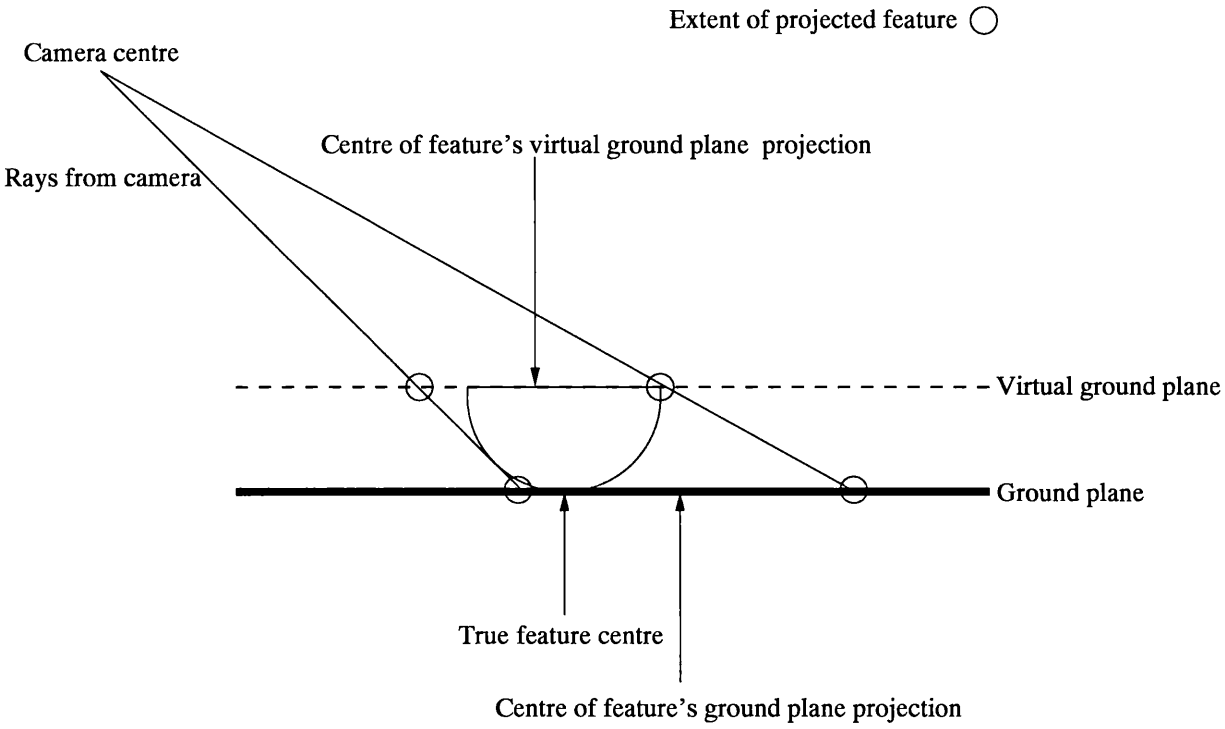


Figure 2.19: Projection errors and the virtual ground plane. Rays intersecting the extremes of the hemispherical feature show the extent of the feature's projection (marked by ○ symbols) on the real and virtual ground planes. The centres of the projected features are also marked, and the projection onto the virtual ground plane is closer to the true object centre than the projection onto the true ground plane.

2.3.9 Image processing – summary

This section has covered the image processing algorithms used to extract features describing plant matter from the images captured by the autonomous vehicle. Two alternative thresholding algorithms have been tested, a fixed threshold algorithm and an adaptive interpolating threshold algorithm, and their performance compared using the area under ROC curves. Operating points have been set using the slope of the ROC curves for the adaptive interpolating threshold algorithm.

Clusters of plant matter pixels ("blobs") are extracted from the images, and the centroid and size of each blob computed. The crop grid model should be fitted to the crop feature centroids, ignoring the weed features. The problem of selecting the correct features to use in fitting the crop grid is a problem of data association, and will be discussed in chapter 6. The blob sizes and centroids are also used in the classification algorithm described in chapter 7 which categorises the blobs as weed or crop for treatment purposes.

We have seen that the image thresholding and pixel clustering processes are subject to certain types of error, although the experimental results presented in chapters 5, 7 and 8 show that these errors do not prevent respectable system performance. Improving the algorithms used for image processing would be a rich source of further research, principally because of the challenging levels of variation in shape, size and texture of the plants and weeds of interest.

2.4 Summary

The autonomous horticultural vehicle operates in a semi-structured field environment, where the available structure arises from the grid formation of the crop plants in the soil. This chapter has introduced the crop grid model which is used as a landmark for navigation that is superior to the row model used previously by Marchant and Brivot [MB95] (appendix A) because it allows the estimation of forward distance in addition to offset and bearing of the crop rows. The method used to estimate the crop grid position is the extended Kalman filter, which is described in chapter 3.

The crop grid model can also be used for crop/weed discrimination. If a set of features that represent plant matter can be extracted from an image, then those features that support the crop grid model may be classified as crop and the remainder as weed. This forms part of a segmentation algorithm described in chapter 7. Plant matter features are extracted using a grey-level thresholding algorithm and a clustering algorithm. Receiver operating characteristic curves were used in the selection of the thresholding algorithm, and also to set the algorithm's operating point. Finally, the limitations of the image processing system were discussed, including feature

localisation errors caused by perspective projection from the image to the ground plane, and the use of a "virtual ground plane" was proposed to reduce the projection errors for crop features.

The crop grid model has been developed, and a method for extracting plant matter features from the image has been demonstrated. The following two chapters detail the Kalman filter estimation mechanism used to track the crop grid model as the vehicle traverses the field.

Chapter 3

Tracking and estimation with Kalman filters

The theme of this thesis is the exploitation of the semi-structured nature of the field environment for the control of a horticultural vehicle. The previous chapter described a model of the structure present in the field, namely the grid pattern formed by the crop plants, together with formulae relating the position of the vehicle to the imaged position of the crop plants. This chapter presents the Kalman filter and the extended Kalman filter, which is used to track the position of the crop pattern through the image sequence captured as the vehicle traverses the field. The Kalman filter is a recursive least-squares estimation algorithm, consisting of a prediction stage followed by a correction stage. Its recursive nature makes it appealing for tracking tasks as information from each image may be utilised as it is collected and the filter does not require a "history" of past inputs to form its latest estimate of the crop grid position. The crop grid structure is used in the *correction* stage where new image measurements are incorporated, whilst knowledge of the vehicle's motion is exploited in the *prediction* stage, which gives the expected position of the grid structure prior to measurement in each image.

The Kalman filter equations are derived below from the conditional density viewpoint, illustrating how the filter forms an optimal estimate of a system's state based upon a set of uncertain measurements and predictions. In addition to the derivation of the Kalman filter, the extended Kalman filter (EKF) is also presented. The EKF is a sub-optimal variant of the Kalman filter adapted for estimation in non-linear systems, such as the perspective imaging process described in the previous chapter. The equations presented here are independent of the crop grid tracking problem. Specific models relating to state evolution and observation for crop grid tracking are given in chapter 4.

Finally, the control theoretic issues of controllability and observability are discussed. The standard tests for the controllability and observability of linear time invariant systems are presented, together with a discussion of their implications for the analysis and design of Kalman filters. A novel test is derived for the observability of the linearised system at the heart of the

extended Kalman filter [SBM98].

The Kalman filter equations and derivation are widely available in the control theory and estimation literature, and we follow Brown and Hwang [BH97], with some reference to Bar-Shalom and Fortmann [BSF88]. Controllability and observability are covered in many standard textbooks, such as Gopal [Gop84] or Jazwinski [Jaz70].

3.1 The Kalman filter

First presented in 1960 by R.E. Kalman [Kal60], the Kalman filter provides a recursive least-squares estimation mechanism for discrete-time systems (an equivalent mechanism for continuous-time systems was published by Kalman and Bucy the following year [KB61]). Given a process whose state at discrete time-step k is denoted by the vector $\mathbf{x}(k)$, a stochastic model for evolution of the process and a method for measuring the process (also stochastic), the Kalman filter forms estimates $\hat{\mathbf{x}}(k)$ of the process state variables by combining predictions from the evolution model with measurements.

For linear systems with Gaussian (normally) distributed stochastic inputs, or noise sources, the Kalman filter is optimal in two senses. First of all it is an *unbiased* estimator, so the estimate is equal to the mean value of the corrupted state. Secondly, the Kalman filter is an *efficient* estimator which produces estimates which minimise the square error between the estimate and the true value. The recursive nature of the filtering algorithm also makes it very convenient to use in real-time systems where data can be integrated into the filter's state estimate as it arrives and there is no need to store a "history" of predictions or measurements. Finally, the filter not only produces state estimates but also a covariance matrix that reflects the confidence in those estimates which may be used in higher-level decision making tasks. For non-linear systems, such as ours, where observations are via a perspective camera, there exists a sub-optimal variant of the Kalman filter known as the extended Kalman filter (EKF) [BSF88]. The EKF uses first order linear approximations of the state evolution and measurement processes to update the state estimate. The noise terms are corrupted by the non-linearity and do not retain their Gaussian form, so the Kalman filter update equations, which are derived using the Gaussian assumption, are no longer optimal. Despite this sub-optimality, the EKF is a popular choice for non-linear estimation problems [LDW91a, SSDW95, MAD99].

The Kalman filter has proved to be immensely popular in the fields of computer vision and robotics for tracking, estimation and data fusion. The short review presented below is far from exhaustive, but aims to provide a broad overview of the uses to which the filter has been put by the computer vision and robotics communities.

Many schemes have used the Kalman filter as an estimation mechanism, for example the tracking of image tokens such as "corners"[1]. Harris [Har92a] and Shapiro *et al* [SWB92] both use corner features tracked by means of Kalman filters to infer 3D descriptions of the moving scene, whilst Deriche and Faugeras [DF90] track line segments extracted from images for the same task. Smith [Smi95b] uses a Kalman filter to track corners in order to extract objects from image sequences.

The line and corner tracking schemes mentioned above all aim to infer structure from image sequences by grouping the tracked tokens into objects. Higher-level descriptions of objects, such as rigid models [Har92b] or contours [CB92] may also be tracked using Kalman filters. In this thesis, the Kalman filter is used to track a model of the crop grid as the vehicle traverses the field. The approach we have adopted is similar to that of Harris' RAPiD tracker [Har92b]. As explained in chapter 2, Harris tracks a rigid wire-frame model of the object of interest, estimating its pose relative to the camera in free space, whilst we are interested in estimating the position of the vehicle relative to the crop grid structure on the ground plane. In the review of model based tracking methods in section 2.1.1 we noted that for tracking more flexible objects, active contour models are often used [KWT87]. The Kalman filter has been used to drive many contour tracking schemes [CB92, TS92, Bau96]. Kalman filters have also found use in parameter estimation problems, such as ellipse fitting [Zha97] and as part of a scheme for homography estimation for an augmented reality application [JZF98] (although its use there is primarily as a tracker), and in stereo matching [PPP+89].

The contour tracking work of Blake *et al* [BCZ93] has been extensively developed to incorporate factors such as learning the dynamics of an object from training sequences [BIR94, NB97], and decoupling changes in contour shape that are caused by change of viewpoint from those caused by an actual change in object shape [RWBM96]. In addition to these improvements to the contour models, a new recursive estimation mechanism has been developed to track the learned contour models through image sequences. This new mechanism is known as the CONDENSATION algorithm [IB96] which, unlike the Kalman filter which assumes Gaussian distributions for its stochastic components, allows the probability distribution for contour position and shape to take an arbitrary form. The strength of the CONDENSATION algorithm is its robustness to noise and distractors in the image. In particular, because it can maintain multi-modal probability distributions, multiple hypotheses of object position may propagated through the tracking

[1] These are not corners in the usual sense of the conjunction of two or more lines, but points in an image that differ from their immediate neighbourhood. The most popular method for corner feature extraction is probably that of Harris and Stephens [HS88].

process. If a uni-modal tracker (such as the Kalman filter) were to be used in such situations, it can lose track fatally because it can only maintain a single hypothesis of object position. The multi-modal tracker allows both distractor and target positions to be represented simultaneously and, once the distractor is found not to satisfy the target dynamics, the true object will re-assert itself as the distribution is propagated through the contour's dynamical model. In our application, the crop grid model is distinctive, and it is unlikely that a group of weeds would mimic the grid formation, so we have opted to use the uni-modal Kalman filter approach which is computationally simpler than the CONDENSATION algorithm.

The Kalman filter (and extended Kalman filter) has also been used in of the robotics applications mentioned in chapter 1, for tasks such as localisation relative to known landmarks or maps [LDW91a, SSDW95], simultaneous map building and localisation [Dav98, Smi95a, Har92a, WNDDW00, LDW91b], and for sensor fusion with dead-reckoning sensors such as odometers, accelerometers and gyroscopes and estimates from global positioning systems [CDW94, Cro95, MAD99].

3.1.1 Filter equations and derivation

As we noted above, the Kalman filter estimates the true value of a process state $\mathbf{x}(k)$ through combination of predicted values of the state and measurement of the state. The equations for performing this combination, and the criteria for optimality are given below, together with a derivation of the equations from the conditional density viewpoint, where the filter's estimate and covariance are presented as a probability density function conditioned on the stream of measurements to date. The derivation presented below closely follows that presented by Brown and Hwang in their excellent introductory text to filtering theory [BH97].

The filter is couched in state-space terms and assumes that the process of interest may be modelled like so:

$$\mathbf{x}(k+1) = \mathbf{A}(k)\mathbf{x}(k) + \mathbf{B}(k)\mathbf{u}(k) + \mathbf{v}(k), \tag{3.1}$$

and it may be measured (or observed) by a system such as

$$\mathbf{z}(k) = \mathbf{H}(k)\mathbf{x}(k) + \mathbf{w}(k), \tag{3.2}$$

where:

$\mathbf{x}(k)$ = $(n \times 1)$ state vector at time step k,

$\mathbf{A}(k)$ = $(n \times n)$ state evolution matrix at time step k, that specifies the transformation of state $\mathbf{x}(k)$ into $\mathbf{x}(k+1)$,

$\mathbf{u}(k)$ = $(p \times 1)$ control input vector; external input which influences the state evolution,

$\mathbf{B}(k)$ = $(n \times p)$ control gain matrix that couples the control inputs to the states,

$\mathbf{v}(k)$ = $(n \times 1)$ noise vector; assumed to be drawn from a white sequence with known covariance,

$\mathbf{z}(k)$ = $(m \times 1)$ observation vector at time step k,

$\mathbf{H}(k)$ = $(m \times n)$ observation matrix; transforms the state vector into the observation space. Describes the ideal (i.e. noise-free) sensing process,

$\mathbf{w}(k)$ = $(m \times 1)$ measurement noise vector, once more assumed to be drawn from a white sequence with known covariance.

The covariance matrices $\mathbf{Q}(k)$ and $\mathbf{R}(k)$ of the noise sources are specified by

$$E[\mathbf{v}(k)\mathbf{v}^T(i)] = \begin{cases} \mathbf{Q}(k), & i = k \\ 0, & i \neq k \end{cases}, \tag{3.3}$$

$$E[\mathbf{w}(k)\mathbf{w}^T(i)] = \begin{cases} \mathbf{R}(k), & i = k \\ 0, & i \neq k \end{cases}, \tag{3.4}$$

$$E[\mathbf{v}(k)\mathbf{w}^T(i)] = 0, \text{for all } k \text{ and } i. \tag{3.5}$$

Equations 3.3 and 3.4 state that the system and observation noise inputs have zero autocorrelation, whilst equation 3.5 defines zero cross-correlation between the two. Deriche and Faugeras [DF90] provide equations for a system with correlated state and observation noise (i.e. where $E[\mathbf{v}(k)\mathbf{w}^T(i)] \neq 0$), although they admit that these correlations are often difficult to model and usually set them to zero.

Equations 3.1 and 3.2 describe the evolution and measurement of the system state respectively. However, the true state $\mathbf{x}(k)$ is unknown. The purpose of the Kalman filter is to recursively estimate the value of this state, starting with an initial estimate[2] and using the state evolution model to *predict* successive values of the state, and measurements to *correct* the predictions. At time step $k+1$, the prediction is denoted $\hat{\mathbf{x}}^-(k+1)$ and is the estimate of the state at time $k+1$ given k previous predict/correct cycles. After incorporation of measurements, the corrected state estimate is written $\hat{\mathbf{x}}(k+1)$ (the $^-$ superscript has been dropped). As noted above,

[2] Initialisation is treated in section 3.3 and chapter 6.

the Kalman filter acts to minimise the error between the estimated state and the true state. The errors at the prediction and correction stages are simply

$$\mathbf{e}^-(k+1) = \mathbf{x}(k+1) - \hat{\mathbf{x}}^-(k+1) \tag{3.6}$$

$$\mathbf{e}(k+1) = \mathbf{x}(k+1) - \hat{\mathbf{x}}(k+1) \tag{3.7}$$

and have associated covariances[3]

$$\mathbf{P}^-(k+1) = E[\mathbf{e}^-(k+1)\mathbf{e}^{-T}(k+1)] = E[(\mathbf{x}(k+1) - \hat{\mathbf{x}}^-(k+1))(\mathbf{x}(k+1) - \hat{\mathbf{x}}^-(k+1))^T] \tag{3.8}$$

$$\mathbf{P}(k+1) = E[\mathbf{e}(k+1)\mathbf{e}^T(k+1)] = E[(\mathbf{x}(k+1) - \hat{\mathbf{x}}(k+1))(\mathbf{x}(k+1) - \hat{\mathbf{x}}(k+1))^T]. \tag{3.9}$$

These covariance matrices reflect the certainty in the state estimate, and are also estimated in the Kalman filtering process.

At time step k, the prediction (or prior) for the state estimate is given (from equation 3.1) by

$$\hat{\mathbf{x}}^-(k+1) = \mathbf{A}(k)\hat{\mathbf{x}}(k) + \mathbf{B}(k)\mathbf{u}(k), \tag{3.10}$$

and the prediction covariance

$$\mathbf{P}^-(k+1) = \mathbf{A}(k)\mathbf{P}(k)\mathbf{A}^T(k) + \mathbf{Q}(k). \tag{3.11}$$

After an observation $\mathbf{z}(k+1)$ has been obtained, the corrected (or posterior) state estimate is calculated as

$$\hat{\mathbf{x}}(k+1) = \hat{\mathbf{x}}^-(k+1) + \mathbf{K}(k+1)(\mathbf{z}(k+1) - \mathbf{H}(k+1)\hat{\mathbf{x}}^-(k+1)), \tag{3.12}$$

with updated error covariance

$$\mathbf{P}(k+1) = [\mathbf{I} - \mathbf{K}(k+1)\mathbf{H}(k+1)]\mathbf{P}^-(k+1), \tag{3.13}$$

where $\mathbf{I}$ is the $n \times n$ identity matrix and $\mathbf{K}(k+1)$ the Kalman gain matrix, given by

$$\mathbf{K}(k+1) = \mathbf{P}^-(k+1)\mathbf{H}(k+1)[\mathbf{H}(k+1)\mathbf{P}^-(k+1)\mathbf{H}^T(k+1) + \mathbf{R}(k+1)]^{-1}. \tag{3.14}$$

The form of equation 3.12 and the gain matrix $\mathbf{K}$ will now be derived.

The mean-square error between the corrected estimate $\hat{\mathbf{x}}(k)$ and the true state $\mathbf{x}(k)$ (given as the covariance matrix $\mathbf{P}(k)$ in equation 3.9) is effectively conditioned on the set of all the

[3] Strictly, these matrices are second moment matrices; however, the errors $\mathbf{e}^-(k)$ and $\mathbf{e}(k)$ are implicitly assumed to have zero mean, in which case the matrices are second *central* moment matrices. However, we shall follow common practice [BSF88] and refer to them as covariance matrices.

measurements $\mathbf{Z}(k) = \{\mathbf{z}(0), \mathbf{z}(1), \ldots, \mathbf{z}(k)\}$ taken up to time step k, i.e. the state estimate $\hat{\mathbf{x}}(k)$ must be considered as the estimate of true state $\mathbf{x}(k)$ *given* the sequence of measurements $\mathbf{Z}(k)$. If we omit the time index k for brevity, this can be written:

$$\begin{aligned} E[(\mathbf{x}-\hat{\mathbf{x}})^T(\mathbf{x}-\hat{\mathbf{x}})|\mathbf{Z}] &= E[(\mathbf{x}^T\mathbf{x} - \mathbf{x}^T\hat{\mathbf{x}} - \hat{\mathbf{x}}^T\mathbf{x} + \hat{\mathbf{x}}^T\hat{\mathbf{x}})|\mathbf{Z}] \\ &= E[\mathbf{x}^T\mathbf{x}|\mathbf{Z}] - E[\mathbf{x}^T|\mathbf{Z}]\hat{\mathbf{x}} - \hat{\mathbf{x}}^T E[\mathbf{x}|\mathbf{Z}] + \hat{\mathbf{x}}^T\hat{\mathbf{x}} \\ &= E[\mathbf{x}^T\mathbf{x}|\mathbf{Z}] + (\hat{\mathbf{x}} - E[\mathbf{x}|\mathbf{Z}])^T(\hat{\mathbf{x}} - E[\mathbf{x}|\mathbf{Z}]) - E[\mathbf{x}^T|\mathbf{Z}]E[\mathbf{x}|\mathbf{Z}], \end{aligned} \tag{3.15}$$

where $\hat{\mathbf{x}}$ is taken out of the expectation operator owing to its functional dependency on $\mathbf{Z}$ (equation 3.12). Equation 3.15 holds the key to the formulation of the Kalman filter. In the last line, only the middle term has a dependency on the filter's estimate $\hat{\mathbf{x}}$, so the best estimate is obtained when this middle term, which is positive semi-definite, is zero, i.e. when

$$\hat{\mathbf{x}}(k) = E[\mathbf{x}(k)|\mathbf{Z}(k)], \tag{3.16}$$

where the time step k has been reinstated. This equation provides a general criteria for a filter that minimises mean-square error. In the Gaussian case, it leads to the formulation of the recursive Kalman filter. Not only does this serve to minimise the mean-square error, but also it guarantees an unbiased state estimate. In fact equation 3.16 states that the estimate is equal to the mean value of the state which is the definition of an unbiased estimator [Spi80].

If the state $\mathbf{x}(k)$ is now considered to be conditioned on the measurement history $\mathbf{Z}(k)$, and a prior optimal estimate (prediction) $\hat{\mathbf{x}}^-(k)$ and its covariance $\mathbf{P}^-(k)$ are available, then, provided the state evolution and observation noise sources are Gaussian, the form of the probability density function for the state $\mathbf{x}(k)$ is

$$f(\mathbf{x}(k)) = N(\hat{\mathbf{x}}^-(k), \mathbf{P}^-(k)), \tag{3.17}$$

where $N(\mathbf{a}, \mathbf{B})$ denotes a normal or Gaussian distribution with mean $\mathbf{a}$ and covariance matrix $\mathbf{B}$. The state and observation are related by equation 3.2, repeated here

$$\mathbf{z}(k) = \mathbf{H}(k)\mathbf{x}(k) + \mathbf{w}(k). \tag{3.18}$$

Thus, given that $\mathbf{w}(k)$ is a zero mean Gaussian process with covariance $\mathbf{R}$ (equation 3.4), the probability density function of the random variable $\mathbf{z}(k)$ may be written

$$f(\mathbf{z}(k)) = N(\mathbf{H}(k)\hat{\mathbf{x}}^-(k), \mathbf{H}(k)\mathbf{P}^-(k)\mathbf{H}^T(k) + \mathbf{R}(k)), \tag{3.19}$$

together with the distribution for $\mathbf{z}(k)$ given full knowledge of $\mathbf{x}(k)$, equation 3.18 also implies that:

$$f(\mathbf{z}(k)|\mathbf{x}(k)) = N(\mathbf{H}(k)\mathbf{x}(k), \mathbf{R}(k)). \tag{3.20}$$

Bayes theorem may now be used to combine the knowledge of the state and observation densities (equations 3.19 and 3.20) to provide an estimate of the state given the measurements $\mathbf{Z}(k)$:

$$f(\mathbf{x}(k)|\mathbf{z}(k)) = \frac{f(\mathbf{z}(k)|\mathbf{x}(k))f(\mathbf{x}(k))}{f(\mathbf{z}(k))}, \tag{3.21}$$

which, if we substitute for the appropriate distributions, becomes

$$f((\mathbf{x}(k)|\mathbf{z}(k)) = \frac{N(\mathbf{H}(k)\mathbf{x}(k), \mathbf{R}(k))N(\hat{\mathbf{x}}^-(k), \mathbf{P}^-(k))}{N(\mathbf{H}(k)\hat{\mathbf{x}}^-(k), \mathbf{H}(k)\mathbf{P}^-(k)\mathbf{H}^T(k) + \mathbf{R}(k))}. \tag{3.22}$$

After some rather involved manipulation (presented in appendix C), it can be seen that this expression is itself a normal distribution with mean

$$\hat{\mathbf{x}}(k) = \hat{\mathbf{x}}^-(k) + \mathbf{P}^-(k)\mathbf{H}^T(k)[\mathbf{H}(k)\mathbf{P}^-(k)\mathbf{H}^T(k) + \mathbf{R}(k)]^{-1}(\mathbf{z}(k) - \mathbf{H}(k)\hat{\mathbf{x}}^-(k)), \tag{3.23}$$

and covariance

$$\mathbf{P}(k) = [(\mathbf{P}^-(k))^{-1} + \mathbf{H}^T(k)\mathbf{R}^{-1}(k)\mathbf{H}(k)]^{-1}. \tag{3.24}$$

Equation 3.23 is the same as the Kalman filter's state correction equation 3.12, with the full expression for the Kalman gain (equation 3.14) inserted. If we apply the matrix inversion lemma (appendix B) to equation 3.24, it yields the Kalman filter state covariance update equation presented earlier (equation 3.13).

Thus, by considering the estimate error to be conditioned on the measurement history, it is possible to derive an estimator for linear systems with Gaussian noise sources which produces estimates with a mean that is both unbiased and also minimises the mean square estimation error (equations 3.15 and 3.16). Inspection of the mean and variance of the estimate conditioned on the measurement history, from combination of prediction and observation using Bayes theorem, yields the Kalman filter state estimate $\hat{\mathbf{x}}(k)$ update and covariance $\mathbf{P}(k)$ update equations respectively.

3.2 The extended Kalman filter

The Kalman filter equations presented so far apply to linear systems with Gaussian noise inputs. However, many systems of practical interest are non-linear, for example the perspective imaging process discussed in chapter 2. In the non-linear case, a technique called the extended Kalman filter (EKF) has been devised [BSF88]. In the EKF, the non-linear state evolution and measurement equations are linearised (typically to the first order) around the predicted state estimate, and then this linearised system is used to compute the Kalman gain for estimate and covariance update. Unlike the linear Kalman filter with Gaussian noise sources, optimality is not guaranteed in the EKF; the Gaussian nature of the noise sources is not preserved by the non-linear transformations in the state evolution and observation equations. With the loss of the Gaussian assumption,

the distributions assumed in equations 3.19, 3.20 and 3.22 are no longer valid, so the gains computed using this assumption will no longer be optimal. Furthermore, in some circumstances, the EKF can diverge from the true state; a poor initial estimate of the state will lead to linearisation of the filter equations about the incorrect point in state space, and inappropriate calculation of the Kalman gain. Estimate updates computed with this erroneous gain may diverge even further from the true state, and the next update will make the situation even worse. However, despite this danger, the EKF has been applied successfully in many practical situations and remains a popular choice of estimator for non-linear systems [LDW91a, SSDW95, MAD99]. This popularity owes much to the EKF's computationally straightforward recursive implementation.

A system with non-linear state and measurement equations may be written

$$\mathbf{x}(k+1) = \mathbf{a}(\mathbf{x}(k), \mathbf{u}(k), k) + \mathbf{v}(k), \tag{3.25}$$

$$\mathbf{z}(k) = \mathbf{h}(\mathbf{x}(k), k) + \mathbf{w}(k), \tag{3.26}$$

where $\mathbf{a}(\cdot)$ is the non-linear state evolution equation and $\mathbf{h}(\cdot)$ the non-linear observation equation. Equation 3.25 is analogous to the linear state evolution equation 3.1, and equation 3.26 to the linear observation equation 3.2. The control input is $\mathbf{u}(k)$, and $\mathbf{w}(k)$ and $\mathbf{v}(k)$ are zero-mean noise sources (with zero autocorrelation and cross-correlation) as earlier (equations 3.3 – 3.5).

Now assume that an estimate $\hat{\mathbf{x}}(k)$ of the true state $\mathbf{x}(k)$ is available. If we use the identity

$$\Delta\mathbf{x}(k) = \mathbf{x}(k) - \hat{\mathbf{x}}^-(k), \tag{3.27}$$

equation 3.25 may be rewritten

$$\hat{\mathbf{x}}^-(k+1) + \Delta\mathbf{x}(k+1) = \mathbf{a}(\hat{\mathbf{x}}^-(k) + \Delta\mathbf{x}(k), \mathbf{u}(k), k) + \mathbf{v}(k), \tag{3.28}$$

and then linearised to first order:

$$\hat{\mathbf{x}}^-(k+1) + \Delta\mathbf{x}(k+1) \approx \mathbf{a}(\hat{\mathbf{x}}^-(k), \mathbf{u}(k), k) + \left[\frac{\partial \mathbf{a}}{\partial \mathbf{x}}\right]_{\mathbf{x}=\hat{\mathbf{x}}^-} \cdot \Delta\mathbf{x} + \mathbf{v}(k). \tag{3.29}$$

The measurement equation may also be couched in terms of $\hat{\mathbf{x}} + \Delta\mathbf{x}$:

$$\mathbf{z}(k) = \mathbf{h}(\hat{\mathbf{x}}^-(k) + \Delta\mathbf{x}(k), k) + \mathbf{w}(k), \tag{3.30}$$

which linearises (to first order) as

$$\mathbf{z}(k) \approx \mathbf{h}(\hat{\mathbf{x}}^-(k), k) + \left[\frac{\partial \mathbf{h}}{\partial \mathbf{x}}\right]_{\mathbf{x}=\hat{\mathbf{x}}^-} \cdot \Delta\mathbf{x} + \mathbf{w}(k). \tag{3.31}$$

Equations 3.29 and 3.31 both contain matrices of partial derivatives,

$$\mathbf{a}_\mathbf{x} = \frac{\partial \mathbf{a}}{\partial \mathbf{x}} = \begin{bmatrix} \frac{\partial a_1}{\partial x_1} & \frac{\partial a_1}{\partial x_2} & \cdots \\ \frac{\partial a_2}{\partial x_1} & \frac{\partial a_2}{\partial x_2} & \cdots \\ \vdots & \vdots & \ddots \end{bmatrix}, \tag{3.32}$$

and

$$\mathbf{h_x} = \frac{\partial \mathbf{h}}{\partial \mathbf{x}} = \begin{bmatrix} \frac{\partial h_1}{\partial x_1} & \frac{\partial h_1}{\partial x_2} & \cdots \\ \frac{\partial h_2}{\partial x_1} & \frac{\partial h_2}{\partial x_2} & \cdots \\ \vdots & \vdots & \ddots \end{bmatrix}, \tag{3.33}$$

respectively. The subscripted a_n and h_n refer to row n of the state evolution function and observation function respectively, and x_n the n^{th} element of the state vector.

Given these linearised equations, update rules for the state estimate and its covariance may be written

$$\hat{\mathbf{x}}(k) = \hat{\mathbf{x}}^-(k) + \mathbf{K}(k)(\mathbf{z}(k) - \mathbf{h}(\hat{\mathbf{x}}^-(k), k)), \tag{3.34}$$

and

$$\mathbf{P}(k) = (\mathbf{I} - \mathbf{K}(k)\mathbf{h}_{\hat{\mathbf{x}}^-(k)})\mathbf{P}^-(k), \tag{3.35}$$

with Kalman gain $\mathbf{K}(k)$ now given by

$$\mathbf{K}(k) = \mathbf{P}^-(k)\mathbf{h}_{\hat{\mathbf{x}}^-(k)}[\mathbf{h}_{\hat{\mathbf{x}}^-(k)}\mathbf{P}^-(k)\mathbf{h}^T_{\hat{\mathbf{x}}^-(k)} + \mathbf{R}(k)]^{-1}. \tag{3.36}$$

The state prediction step is handled as follows

$$\hat{\mathbf{x}}^-(k+1) = \mathbf{a}(\hat{\mathbf{x}}(k), \mathbf{u}(k), k), \tag{3.37}$$

and state covariance prediction by

$$\mathbf{P}^-(k+1) = \mathbf{a}_{\hat{\mathbf{x}}^-(k)}\mathbf{P}(k)\mathbf{a}^T_{\hat{\mathbf{x}}^-(k)} + \mathbf{Q}(k), \tag{3.38}$$

which is analogous to equation 3.11 in the linear Kalman filter. Note that the state prediction equation 3.37 retains the non-linear form of the state evolution equation 3.25, whilst equation 3.38 uses the approximate linearised system to propagate the state covariance matrix forward in time.

3.3 Practicalities – initialisation and data association

Two issues that are important for practical Kalman filtering applications are initialisation, which was mentioned above, and data association. These are covered in depth in chapter 6 but merit coverage here as a pre-cursor to the experiments presented in chapter 5. As noted above, estimates produced by the extended Kalman filter are not guaranteed to converge to the true value of the state, and poor initialisation can lead to divergence from the onset of tracking.

Data association is the task of selecting observed features for incorporation into the filter's estimate. The measurement equation 3.2 states that the observation of $\mathbf{z}(k)$ is a transformed version of the state corrupted by additive Gaussian noise. This implies a single observation is

available which corresponds to the true state. We know from chapter 2 that our image processing algorithms do not extract crop plants alone, but both crop and weed features, together with some soil pixels. Because we have both "target" crop plant features and "clutter" weed features, there is no longer a simple one-to-one correspondence between observed features and the state estimate, so some method to pair observations with predicted feature positions is required[4]. This is the task of data association and is vital to successful estimation.

In chapters 4 and 5 we give details of the process and measurement models used in the crop grid tracker, and results from initial experiments with the tracker. These results could not have been obtained without a good estimate of the initial state and a data association policy, and algorithms fulfilling both of these roles are presented in chapter 6.

3.4 Controllability and observability

As we have noted, the Kalman filter has proved to be a popular tool in computer vision [DF90, Har92a, SWB92, Har92b, BCZ93]. In contrast to the widespread use of the Kalman filter, two issues raised by Kalman, *controllability* and *observability*, are seldom seen in the machine vision literature, although Wildenberg [Wil97] has shown their value as a design aid for linear time-invariant Kalman filters. The two concepts will first be defined for linear time-invariant (LTI) systems and their implications for Kalman filtering discussed. In this discussion, a new term *corruptibility* is introduced as a parallel concept for controllability relating to noise inputs.

Finally, a novel test for the observability of an extended Kalman filter which has linear state evolution equations with a non-linear measurement model is derived. Extended Kalman filters with this mixed linear/non-linear form are of particular interest in this thesis, as will become clear in the next chapter. The new formulation of the observability test is required for the analysis of such mixed filters.

3.4.1 Controllability and observability in LTI systems

The definitions and results given here for controllability and observability of linear systems may be found in many standard textbooks [Gop84, Jaz70]. In the discussion below, the linear, discrete time invariant, systems are given by

$$\mathbf{x}(k+1) = \mathbf{A}\mathbf{x}(k) + \mathbf{B}\mathbf{u}(k), \tag{3.39}$$

$$\mathbf{z}(k) = \mathbf{H}\mathbf{x}(k), \tag{3.40}$$

which are deterministic systems analogous to the stochastic systems in equations 3.1 and 3.2.

[4]Even if weed features were not observed, data association would be needed to match observed crop features with their corresponding positions in the crop grid.

3.4.2 Controllability

A system is said to be *controllable* if every state vector $\mathbf{x}(k)$ can be transformed to a desired state in finite time by the application of unconstrained control inputs $\mathbf{u}(k)$ [Gop84]. Evidently then, an *un*controllable system is one where some elements of the state vector $\mathbf{x}(k)$ cannot be affected by the control input.

The test for controllability of a linear time-invariant system is given by:

A system with state vector $\mathbf{x}$ of dimension n, is controllable if the *controllability matrix*

$$\mathcal{C} = [\mathbf{B}, \mathbf{AB}, \ldots, \mathbf{A}^i\mathbf{B}, \ldots, \mathbf{A}^{n-1}\mathbf{B}], \tag{3.41}$$

has column rank n (i.e. n linearly independent columns).

The proof of this statement is based on successive substitution of the state evolution equation (equation 3.39) into itself to find a solution for $\mathbf{x}(N)$ in terms of an original state $\mathbf{x}(0)$ and a series of control inputs $\mathbf{u}(0) \ldots \mathbf{u}(N-1)$.

3.4.3 Observability

A system is said to be *observable* at a time step k_0 if, for a state $\mathbf{x}(k_0)$ at that time, there is a finite $k_1 > k_0$ such that knowledge of the outputs $\mathbf{z}$ from k_0 to k_1 are sufficient to fully determine the state $\mathbf{x}(k_0)$. Thus it is evident that an *un*observable system is one where the values of some elements in the state vector at time k_0 may not be determined from examination of the system output regardless of the number of observations taken. That observability is specified over an interval highlights the fact that whilst a single observation of the system at time k may not be enough to obtain the complete state, additional observations may allow the full state information to be accumulated. Evidently, for time invariant systems, the time k_0 is unimportant.

For linear time invariant systems, the test for observability is given by:

A system with state vector $\mathbf{x}$ of dimension n is observable if the *observability matrix*

$$\mathcal{O} = \begin{bmatrix} \mathbf{H} \\ \mathbf{HA} \\ \vdots \\ \mathbf{HA}^i \\ \vdots \\ \mathbf{HA}^{n-1} \end{bmatrix}, \tag{3.42}$$

has row rank n (i.e. n linearly independent rows).

The proof of this test [Gop84] uses the state evolution and observation equations (3.39 and 3.40) to determine the value of $\mathbf{z}(k)$ for $0 \leq k \leq n-1$ in terms of $\mathbf{x}(0)$ and the known control inputs $\mathbf{u}(k)$ during that time period. It is then straightforward to show that $\mathbf{x}(0)$ can be completely determined (and is therefore observable) if the matrix $\mathcal{O}$ has full row rank.

3.4.4 Implications for Kalman filtering

The above tests for controllability and observability may also be applied to stochastic systems, whose evolution and observation models are written:

$$\mathbf{x}(k+1) = \mathbf{A}\mathbf{x}(k) + \mathbf{B}\mathbf{u}(k) + \mathbf{V}\mathbf{v}(k), \tag{3.43}$$

$$\mathbf{z}(k) = \mathbf{H}\mathbf{x}(k) + \mathbf{w}(k). \tag{3.44}$$

These equations are repeated from equations 3.1 and 3.2, with a slight change of notation. The system Gaussian noise input $\mathbf{v}(k)$, which was described as having zero mean and covariance $\mathbf{Q}(k)$ in section 3.1.1 is now considered to have zero mean and unit variance. The matrix $\mathbf{V}$ describes the gains applied to the unit noise source $\mathbf{v}(k)$, and stipulates how the noise affects the state evolution; the covariance of the noise influencing the state evolution is $\mathbf{Q} = \mathbf{V}^T\mathbf{V}$. The noise terms may simply be treated as another control input, and the controllability test described above (equation 3.41) may be applied, using matrix $\mathbf{V}$ in place of $\mathbf{B}$; the result indicates the ability of the noise to affect the state. The additive zero-mean Gaussian noise term $\mathbf{w}$ in (3.44) clearly does not affect the measure of observability, which is solely dependent on $\mathbf{H}$ and $\mathbf{A}$. The noise source does, however affect the meaning of observability. If the deterministic system of equations 3.39 and 3.40 is observable, exact information about the state $\mathbf{x}$ may be derived from the measurements $\mathbf{z}$. If the stochastic system of equations 3.43 and 3.44 is observable, it means that the uncertain estimate $\hat{\mathbf{x}}(k)$ of the exact state $\mathbf{x}(k)$ can be determined from the measurements $\mathbf{z}(k)$.

The Kalman filter prediction equations are written here again; these will be used to illustrate the implications of controllability and observability in Kalman filter performance.

$$\hat{\mathbf{x}}^-(k+1) = \mathbf{A}\hat{\mathbf{x}}(k) + \mathbf{B}\mathbf{u}(k), \tag{3.45}$$

$$\mathbf{P}^-(k+1) = \mathbf{A}\mathbf{P}(k)\mathbf{A}^T + \mathbf{Q}. \tag{3.46}$$

$\mathbf{Q}$ is the covariance matrix of the system noise, i.e. $\mathbf{V}^T\mathbf{V}$. This is also independent of time.

3.4.5 Controllability

If the pair **A** and **B** form an uncontrollable system, this has no implications for the Kalman filter other than that the control inputs **u** will not affect every element of the state estimate $\hat{\mathbf{x}}$ during the prediction step. This will not affect tracking performance unless, of course, it reveals a flaw in the modelling of the system dynamics. However, if the system formed by **A** and **V** is uncontrollable, this means that the Gaussian noise sources in **v** do not affect all of elements of the state, i.e. some state elements are uncorrupted by the system noise. The diagonal elements of **P** corresponding to these "incorruptible" states will be driven to zero by the Kalman filter (which minimises the trace of **P**), and once this has happened, the estimates of these states are fixed; no further observations will alter their values. To test whether this collapse of variance will occur (i.e. whether noise can affect all states), the following test of *corruptibility* may be used:

A system with state vector **x** of dimension n, will be fully corruptible if the *corruptibility matrix*

$$\mathcal{C}_v = [\mathbf{V}, \mathbf{AV}, \ldots, \mathbf{A}^i\mathbf{V}, \ldots, \mathbf{A}^{n-1}\mathbf{V}], \tag{3.47}$$

has column rank n (i.e. n linearly independent columns).

This test is analogous to the controllability test for the **A**, **B** pair (equation 3.41), and is derived in a similar manner, by successively substituting equation 3.43 into itself to produce an expression for $\mathbf{x}(n-1)$ in terms of a series control inputs $\mathbf{u}(k)$ and noise inputs $\mathbf{v}(k)$. The term corruptibility has been introduced here simply to distinguish between the ability of noise and control inputs to influence the state variables; the controllability test may be applied independently to the **A**, **B** pair of equation 3.43, and the corruptibility test to the **A**, **V** pair. Wildenberg [Wil97] also tests whether a noise source can affect all state variables using the controllability test; in Wildenberg's filters, the only inputs modelled are noise, so the controllability/corruptibility distinction is not required.

Although such an approach may at first seem a little curious, there is sometimes good reason to allow the variance of an estimate to collapse to zero. Reynard *et al* [RWBM96] estimate the mean of a process; clearly a mean is a single, fixed quantity which does not vary with time (it is statistically stationary). If the estimation process allows accurate evaluation of the mean it is quite correct to use the process covariance to constrain the value of the estimated mean by allowing the estimate variance to reduce to zero (the issue of controllability of this system is discussed in depth in the later work by Wildenberg [Wil97]). The state describing the process mean

is incorruptible whilst the other, dynamic, states are corruptible, as is usual for all states in most Kalman filters. In the work of Waite *et al* [WOFH93], where the covariance matrix is used to provide partial constraints on aligning patches of range data, it is noted that in certain degenerate cases there are problems with their algorithm, although this is not named as a controllability or corruptibility problem.

Inspection of equation 3.46, which governs the evolution of the state covariance **P**, will confirm that certain diagonal elements cannot increase (and hence the mean-square error on the corresponding state estimate cannot increase) if the **A**,**V** pair are uncontrollable. The first term in the equation, $\mathbf{A}\mathbf{P}(k)\mathbf{A}^T$ describes the process of noise transfer between states during state evolution – by definition, in an uncontrollable system noise cannot transfer into any uncontrollable state via this mechanism, therefore it cannot be responsible for any increase in the variance of uncontrollable states. The second term in (3.46) is the covariance $\mathbf{Q}$ $(= \mathbf{V}^T\mathbf{V})$ which represents the direct noise input; evidently in an incorruptible system this contribution to **P** must be zero for any incorruptible states (this follows from the form of **V**, which *must* have zero elements in the row corresponding to the incorruptible state, otherwise noise is being directly input to that state at each iteration). If the diagonal elements of **P** corresponding to incorruptible states cannot increase, the Kalman filter will drive these elements toward zero, and new observations will have negligible influence on the estimate of the incorruptible states.

3.4.6 Observability

Whilst incorruptible systems are sometimes desirable for Kalman filtering, a Kalman filter built around a system with unobservable states will simply not work. By definition, an unobservable state is one about which no information may be obtained through the observation equations. In the absence of information, the filter's estimate for that state will not converge on a meaningful solution.

3.5 Corruptibility, observability and the EKF

The controllability and observability conditions for linear systems are well known in control theory, but in general no similar tests are available for discrete-time non-linear systems (although concepts such as local controllability are known for some continuous-time non-linear systems [OB97]). The extended Kalman filter, however, approximates the underlying non-linear system with a series of linear approximations. It is the controllability and observability of this linearised system that is important. Of course, if the linearisation does not reflect the underlying non-linear system accurately enough, then these tests will be meaningless. However, if the approximated linearised system is a poor reflection of the underlying non-linear system then the

extended Kalman filter will in any case be unlikely to converge on a respectable estimate regardless of controllability, corruptibility and observability issues.

As will be seen in the next chapter, the filter of interest in this thesis has a linear state evolution model, together with non-linear observation equations:

$$\mathbf{x}(k+1) = \mathbf{A}\mathbf{x}(k) + \mathbf{B}\mathbf{u}(k) + \mathbf{V}\mathbf{v}(k), \tag{3.48}$$

$$\mathbf{z}(k) = \mathbf{h}(\mathbf{x}(k), k) + \mathbf{w}(k). \tag{3.49}$$

The corruptibility of this system may be determined using the rank of the matrix $\mathcal{C}_v$ (equation 3.47), whilst a new method must be devised to determine the observability of the linearised observation system.

Recall from above that by putting

$$\mathbf{h}(\mathbf{x}(k)) = \mathbf{h}(\hat{\mathbf{x}}(k) + \Delta\mathbf{x}(k)), \tag{3.50}$$

where

$$\Delta\mathbf{x}(k) = \mathbf{x}(k) - \hat{\mathbf{x}}(k), \tag{3.51}$$

the non-linear function $\mathbf{h}(\mathbf{x}(k))$ may be linearised about the estimated $\hat{\mathbf{x}}(k)$ using a first-order Taylor series expansion:

$$\mathbf{h}(\hat{\mathbf{x}}(k)) \approx \mathbf{h}(\hat{\mathbf{x}}(k)) + \mathbf{h}_{\hat{\mathbf{x}}(k)}\Delta\mathbf{x}(k), \tag{3.52}$$

where $\mathbf{h}_{\hat{\mathbf{x}}(k)}$ is the matrix of partial derivatives of $\mathbf{h}(\mathbf{x}(k))$ with respect to the elements of $\mathbf{x}(k)$ evaluated at the estimated $\hat{\mathbf{x}}(k)$. If we use this linear approximation, equation 3.49 may be rewritten as:

$$\mathbf{z}(k) = \mathbf{h}(\hat{\mathbf{x}}(k)) + \mathbf{h}_{\hat{\mathbf{x}}(k)}\Delta\mathbf{x}(k) = \mathbf{h}(\hat{\mathbf{x}}(k)) + \mathbf{h}_{\hat{\mathbf{x}}(k)}(\mathbf{x}(k) - \hat{\mathbf{x}}(k)). \tag{3.53}$$

Given that the form of $\mathbf{h}$ and $\mathbf{h}_{\hat{\mathbf{x}}(k)}$, together with the value of $\hat{\mathbf{x}}(k)$ are known, equation 3.53 may be rewritten as a sum of known estimated and unknown parts

$$\mathbf{z}(k) = (\mathbf{h}(\hat{\mathbf{x}}(k)) - \mathbf{h}_{\hat{\mathbf{x}}(k)}\hat{\mathbf{x}}(k)) + \mathbf{h}_{\hat{\mathbf{x}}(k)}\mathbf{x}(k) = \mathbf{F}(\hat{\mathbf{x}}(k)) + \mathbf{h}_{\hat{\mathbf{x}}(k)}\mathbf{x}(k). \tag{3.54}$$

If we use equations 3.48 and 3.54, we can now derive an expression for the observability matrix $\mathcal{O}_{lin}$ for this linearised system. Thus, if we start arbitrarily at time $k = 0$, a series of expressions for $\mathbf{z}(k)$ may be derived:

$$\begin{aligned} \mathbf{z}(0) &= \mathbf{F}(\hat{\mathbf{x}}(0)) + \mathbf{h}_{\hat{\mathbf{x}}(0)}\mathbf{x}(0) \\ \mathbf{z}(1) &= \mathbf{F}(\hat{\mathbf{x}}(1)) + \mathbf{h}_{\hat{\mathbf{x}}(1)}\mathbf{x}(1) \end{aligned} \tag{3.55}$$

$$= \mathbf{F}(\hat{\mathbf{x}}(1)) + \mathbf{h}_{\hat{\mathbf{x}}(1)}[\mathbf{A}\mathbf{x}(0) + \mathbf{B}\mathbf{u}(0)] \quad (3.56)$$

$$\mathbf{z}(2) = \mathbf{F}(\hat{\mathbf{x}}(2)) + \mathbf{h}_{\hat{\mathbf{x}}(2)}\mathbf{x}(2)$$

$$= \mathbf{F}(\hat{\mathbf{x}}(2)) + \mathbf{h}_{\hat{\mathbf{x}}(2)}[\mathbf{A}^2\mathbf{x}(0) + \mathbf{A}\mathbf{B}\mathbf{u}(0) + \mathbf{B}\mathbf{u}(1)] \quad (3.57)$$

$$\vdots$$

$$\mathbf{z}(n-1) = \mathbf{F}(\hat{\mathbf{x}}(n-1)) + \mathbf{h}_{\hat{\mathbf{x}}(n-1)}\mathbf{A}^{n-1}\mathbf{x}(0)$$

$$+ \mathbf{h}_{\hat{\mathbf{x}}(n-1)}[\mathbf{A}^{n-2}\mathbf{B}\mathbf{u}(0) + \cdots + \mathbf{A}\mathbf{B}\mathbf{u}(n-2) + \mathbf{B}\mathbf{u}(n-1)], \quad (3.58)$$

where n is the dimension of the state vector $\mathbf{x}(k)$. Equations 3.55 – 3.58 may be re-arranged into a matrix-vector form as follows

$$\begin{bmatrix} \mathbf{h}_{\hat{\mathbf{x}}(0)} \\ \mathbf{h}_{\hat{\mathbf{x}}(1)}\mathbf{A} \\ \vdots \\ \mathbf{h}_{\hat{\mathbf{x}}(n-1)}\mathbf{A}^{n-1} \end{bmatrix} \mathbf{x}(0) = \begin{bmatrix} \mathbf{z}(0) - \mathbf{F}(\hat{\mathbf{x}}(0)) \\ \mathbf{z}(1) - \mathbf{F}(\hat{\mathbf{x}}(1)) - \mathbf{h}_{\hat{\mathbf{x}}(1)}\mathbf{B}\mathbf{u}(0) \\ \vdots \\ \mathbf{z}(n-1) - \mathbf{F}(\hat{\mathbf{x}}(n-1)) - \mathbf{h}_{\hat{\mathbf{x}}(n-1)}[\mathbf{A}^{n-2}\mathbf{B}\mathbf{u}(0) + \cdots \\ \qquad + \mathbf{A}\mathbf{B}\mathbf{u}(n-2) + \mathbf{B}\mathbf{u}(n-1)] \end{bmatrix}. \quad (3.59)$$

For $\mathbf{x}(0)$ to be determined from equation 3.59, it can be seen that the matrix on the left hand side of the equation must have row rank n, where n is the dimension of $\mathbf{x}$. This matrix is the equivalent of the observability matrix $\mathcal{O}$ given in equation 3.42. The similarities in structure stand out; in place of the linear observation matrices $\mathbf{H}$ are the matrices of partial derivatives of the observation function at a series of time steps k. This allows us to state an observability condition for the linearised system as follows:

A system with state vector $\mathbf{x}$ of dimension n and a non-linear observation model linearised at successive states $\hat{\mathbf{x}}(k)$ is observable if the *linearised observability matrix*

$$\mathcal{O}_{lin} = \begin{bmatrix} \mathbf{h}_{\hat{\mathbf{x}}(0)} \\ \mathbf{h}_{\hat{\mathbf{x}}(1)}\mathbf{A} \\ \vdots \\ \mathbf{h}_{\hat{\mathbf{x}}(i)}\mathbf{A}^{i} \\ \vdots \\ \mathbf{h}_{\hat{\mathbf{x}}(n-1)}\mathbf{A}^{n-1} \end{bmatrix}, \quad (3.60)$$

has row rank n (i.e. n linearly independent rows) regardless of the values of $\hat{\mathbf{x}}(k)$.

If the rank of $\mathcal{O}_{lin}$ *is* dependent on the value of the state vector $\hat{\mathbf{x}}(\mathrm{k})$, then it may be possible to detect conditions where the system becomes unobservable, at which point observations should not be used to update the filter estimates. If it is possible to show that, regardless of the value of

$\hat{\mathbf{x}}(k)$, the matrix $\mathbf{h}_{\hat{\mathbf{x}}(k)}$ has rank greater than or equal to the dimension of the state vector, then evidently the matrix $\mathcal{O}_{lin}$ will also satisfy the rank condition and the linearised system will be observable.

3.6 Summary

The linear estimator known as the Kalman filter has been described, and its variant for non-linear systems, the extended Kalman filter, presented. The Kalman filter enables estimation of the state variable and its covariance to be calculated recursively and guarantees unbiased minimum mean-square error estimates in linear systems with Gaussian noise sources. Although the extended Kalman filter cannot offer the same guarantees, it is a useful tool for non-linear estimation and retains the recursive computational structure of the Kalman filter that makes it convenient for real-time tasks such as tracking the crop grid structure from an autonomous horticultural vehicle.

In addition to developing the Kalman filter, R.E. Kalman also introduced the concepts of controllability and observability to aid the analysis of linear state-space systems. Although these contributions are less widely mentioned in the vision literature, they can be useful in predicting certain aspects of a Kalman filter's behaviour. Controllability implies that every element of the filter's state vector will be affected by system control inputs. An analogous concept of corruptibility has been proposed which determines whether each element of the filter's state vector may be affected by system noise. If a state is incorruptible, the estimate variance will be driven to zero by the filter, and further measurements of this state will not affect the filter's estimate. It has been noted that in some cases, incorruptibility is a desirable feature. However, if a state is unobservable, then the filter will simply not be able to estimate the whole of that state. At least one component of any estimate it produces will be completely unreliable. The system of interest to this thesis has a linear state evolution model, with non-linear observation model (the perspective camera). To test the observability of this system a novel condition has been derived using the linearised observation equations from the extended Kalman filter.

The next chapter describes the state evolution and observation equations for the filters designed to track the crop grid structure, both off-line on a desktop workstation during system development and on-line on the autonomous vehicle itself during systems testing and final evaluation. The tests for corruptibility and observability are put into practice to check the viability of these filters prior to their implementation, which is detailed in chapter 5.

Chapter 4

Process and observation models for crop grid tracking

Now that the extended Kalman filter has been introduced, together with tests for its controllability, corruptibility and observability, it is appropriate to specify the state evolution and observation models for the crop grid tracking problem that is of central concern to this thesis. This chapter outlines evolution and observation models for filters that run off-line on digitised sequences captured from the vehicle, and also the vision system filter that runs on-line on the autonomous vehicle. Experimental trials carried out with these filters are detailed in chapter 5.

Sections 4.1 – 4.5 focus on the analysis and explanation of the off-line filters. These have been designed to test the principle of crop grid tracking off-line before full implementation on the autonomous vehicle. Two off-line filters are discussed; in the first, the crop grid spacing parameters $\bar{r}$ and $\bar{l}$[1] are held at fixed values, and in the second they are estimated by the filter. The state evolution and observation models are described for each filter separately. In both cases there is a linear state evolution model and non-linear observation model, the non-linearity arising from the perspective projection between the ground plane of the plants and the image plane of the vehicle's camera.

The corruptibility and observability of these two filters is examined using the tests developed in the previous chapter [SBM98]. To aid the observability analysis, two alternative (but equivalent) formulations of the Kalman filter algorithm are presented. The first is the information, or inverse covariance filter. Its equivalence with the Kalman filter equations of chapter 3 is demonstrated and then it is used to develop a parallel update filter (section 4.3.2). This parallel update filter incorporates several observations simultaneously, in contrast with the sequential update of the standard Kalman filter equations, and as such it seems a more "natural" candidate for use on a set of observations that are extracted from a single image. The two formulations

[1] These dictate the space between crop rows ($\bar{r}$) and within each row ($\bar{l}$). See chapter 2 and figure 4.1 for details.

of Kalman filter are mathematically identical, but in the parallel update filter, the conditions for observability are more easily explained (and they hold for any equivalent serial implementation of the Kalman filtering algorithm).

Finally, section 4.6 describes the on-line system architecture designed by Hague for the autonomous vehicle [SHMB99]. This architecture makes use of a third alternative Kalman filter formulation, the data compression filter[2], to integrate the crop grid tracker with the vehicle's dead-reckoning system. In the on-line architecture, the vision system is essentially treated as an intelligent sensor. The dead-reckoning system (chapter 1) sends a state prediction to the vision system which compresses the individual crop feature observations from the image into a single "pseudo-observation" with accompanying covariance matrix. These are passed back to the dead-reckoning system which treats them as a normal measurement (with covariance) to be used in the state estimate update.

4.1 State evolution model

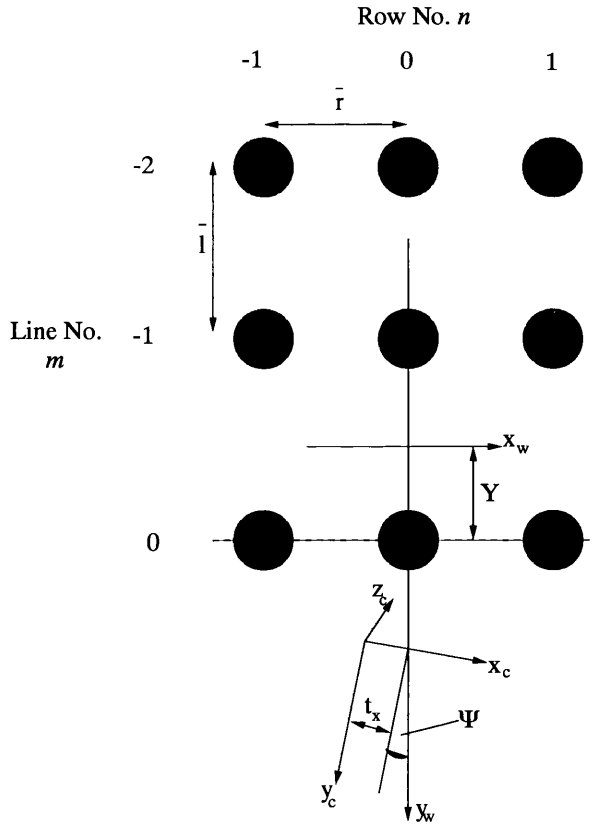


Figure 4.1: The grid model.

Figure 4.1 (repeated from chapter 2) shows the schematic view of a patch of crop. The model of the crop comprises the two grid spacing parameters $\bar{r}$ and $\bar{l}$, and the position of the ve-

[2]Again, the data compression filter is mathematically equivalent to the Kalman filter of chapter 3.

hicle relative to the crop is specified by the lateral offset t_x, forward distance Y and bearing angle Ψ. Two different filters are described here; in the first, the parameters $\bar{r}$ and $\bar{l}$ are assumed to be constant, and are not estimated by the filter. In the second $\bar{r}$ and $\bar{l}$ are variable and estimated by the filter as it traverses the field. Variation of $\bar{r}$ and $\bar{l}$ allows compensation for local disturbances in the planting pattern. The state evolution models for each case are given separately below.

4.1.1 The model with fixed grid parameters

In this filter, the state vector $\mathbf{x}(k)$ comprises the three vehicle position variables $\mathbf{x}(k) = [t_x(k), Y(k), \Psi(k)]^T$. The equations describing their evolution between time k and $k+1$ are

$$\mathbf{x}(k+1) = \mathbf{I}_3\mathbf{x}(k) + \mathbf{I}_3 \begin{bmatrix} U_{t_x} \\ U_Y \\ U_\Psi \end{bmatrix} + \mathbf{V}\mathbf{v}(k), \tag{4.1}$$

where $\mathbf{I}_3$ is the 3×3 identity matrix and the subscripted U values are control inputs relating to the vehicle's motion between image frames. The noise gain matrix $\mathbf{V}$ is diagonal, such that

$$\mathbf{V}^T(k)\mathbf{V}(k) = \mathbf{Q}(k) = \begin{bmatrix} \sigma^2_{t_x}(k) & 0 & 0 \\ 0 & \sigma^2_Y(k) & 0 \\ 0 & 0 & \sigma^2_\Psi(k) \end{bmatrix}. \tag{4.2}$$

The σ^2 terms relate to the mean-square error introduced at each prediction step.

It should be noted that the simple model of equation 4.1 does not reflect the non-linear kinematics of the vehicle, which are modelled more precisely by Hague and Tillett [HT96] as follows:

$$\begin{bmatrix} t_x(k+1) \\ Y(k+1) \\ \Psi(k+1) \end{bmatrix} = \begin{bmatrix} t_x(k) + \frac{u_1(k)+u_2(k)}{2}\sin\Psi(k) \\ Y(k) + \frac{u_1(k)+u_2(k)}{2}\cos\Psi(k) \\ \Psi(k) + \frac{u_2(k)-u_1(k)}{W} \end{bmatrix} + \mathbf{V}(k)\mathbf{v}(k). \tag{4.3}$$

Where u_1 and u_2 are the incremental motions of the left and right driven wheels respectively and W is the width of the vehicle between the driven wheels. Equation 4.3 holds if the vehicle is on a trajectory of low curvature. A different model holds for higher curvature travel such as the headland turn executed at the end of each row of crop, which may also be found in Hague and Tillett [HT96]. This thesis concentrates on low curvature travel along the crop rows, so the other motion models are of only incidental interest.

Unfortunately, owing to hardware limitations, it has not been possible to collect image data from the vehicle that can be temporally aligned with logged wheel velocities, so the model of equation 4.3 cannot be used off-line. However, given small values of $\Psi(k)$, such that $\cos\Psi \approx 1$ and $\sin\Psi \approx 0$, and equal wheel velocities u, it can be seen that equations 4.1 and 4.3 are

approximately equal, with $U_{t_x} = 0$, $U_Y = u$ and $U_\Psi = 0$. Furthermore, the differences may be compensated for to some extent by the noise terms $\mathbf{V}(k)\mathbf{v}(k)$ in equation 4.1, especially when the vehicle is travelling along a row of crop and U_{t_x} and U_Ψ tend to be small corrections not dissimilar to random inputs.

4.1.2 The model with estimated grid parameters

When the grid parameters $\bar{r}$ and $\bar{l}$ are variables to be estimated, the filter state vector becomes $\mathbf{x}(k) = [t_x(k), Y(k), \Psi(k), \bar{r}(k), \bar{l}(k)]^T$, and the equation describing the evolution between time k and $k+1$ is

$$\mathbf{x}(k+1) = \mathbf{I}_5\mathbf{x}(k) + \mathbf{I}_5 \begin{bmatrix} U_{t_x} \\ U_Y \\ U_\Psi \\ 0 \\ 0 \end{bmatrix} + \mathbf{V}\mathbf{v}(k), \tag{4.4}$$

where $\mathbf{I}_5$ is the 5×5 identity matrix and the subscripted U values are control inputs relating to the vehicle's motion between image frames. It should be noted that, except for random variations due to noise, the mean plant spacings $\bar{r}$ and $\bar{l}$ are assumed to be constant. The noise gain matrix is diagonal, such that

$$\mathbf{V}^T(k)\mathbf{V}(k) = \mathbf{Q}(k) = \begin{bmatrix} \sigma^2_{t_x}(k) & 0 & 0 & 0 & 0 \\ 0 & \sigma^2_Y(k) & 0 & 0 & 0 \\ 0 & 0 & \sigma^2_\Psi(k) & 0 & 0 \\ 0 & 0 & 0 & \sigma^2_{\bar{r}}(k) & 0 \\ 0 & 0 & 0 & 0 & \sigma^2_{\bar{l}}(k) \end{bmatrix}. \tag{4.5}$$

The σ^2 terms relate to the mean-square error introduced at each prediction step. For t_x, Y and Ψ these are fixed, but for $\bar{r}$ and $\bar{l}$ the σ^2 terms are non-zero *only* when a new line of plants come into view or when a line of plants go out of view as the vehicle travels along the field. The reason for this is that process noise $\mathbf{Q}$ should be added only when the parameters being estimated change, and for the estimated mean values of the plant spacing parameters, this only happens when new plants appear in the image, or previously seen plants move out of the field of view.

4.1.3 Forward distance estimation

The forward distance measurement Y illustrated in figure 4.1 describes the distance on the ground plane between the centre of the image and the crop plant nearest the bottom of the image. As the vehicle progresses along the field, this plant will eventually pass out of the image and will no longer be observed; the next plant along the row will now be nearest to the bottom of the image.

To keep the Y estimate consistent with the position of the bottom-most line of crop plants in the image[3], each time a line of plants passes out of the bottom of the image, the point of reference for Y is moved to the plant which is now bottom-most in the image. If this event occurs at time step k' the change of reference is effected as follows:

$$Y(k'+1) = Y(k') + \bar{l}(k') + U_Y(k'). \tag{4.6}$$

Note that in the case of the filter where $\bar{r}$ and $\bar{l}$ are kept fixed, $\bar{l}(k)$ is obviously constant. $U_Y(k')$ is the control input change in Y caused by vehicle motion between time k' and $k'+1$. In terms of the state evolution models, equation 4.6 implies subtle changes, depending on whether the grid parameters $\bar{r}$ and $\bar{l}$ are estimated or not. These changes are detailed below.

4.1.4 Forward distance estimation with fixed grid parameters

In this case, the state evolution model of equation 4.1 remains unchanged, except that, if a line of plants passes out of the bottom of the image at time step k',

$$\mathbf{x}(k'+1) = \mathbf{I}_3\mathbf{x}(k') + \begin{bmatrix} U_{t_x} \\ U_Y + \bar{l} \\ U_\Psi \end{bmatrix} + \mathbf{V}\mathbf{v}(k'), \tag{4.7}$$

i.e. the change of reference point for Y is treated as a control input.

4.1.5 Forward distance estimation with estimated grid parameters

When the grid parameters are estimated, the quantity $\bar{l}(k)$ is part of the state vector, so equation 4.6 implies a change in the matrix $\mathbf{A}$. If a line of plants passes out of the bottom of the image at time step k', the state evolution model becomes

$$\mathbf{x}(k'+1) = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \mathbf{x}(k') + \begin{bmatrix} U_{t_x} \\ U_Y \\ U_\Psi \\ 0 \\ 0 \end{bmatrix} + \mathbf{V}\mathbf{v}(k). \tag{4.8}$$

By comparing equation 4.8 with equation 4.4 it can be seen that when the grid parameters are estimated, the increment of Y requires an alteration of the state evolution matrix $\mathbf{A}(k)$. The new matrix $\mathbf{A}(k')$ will not only affect the predicted value $\hat{\mathbf{x}}^-(k'+1)$ as desired, but also change the prediction covariance matrix $\mathbf{P}^-(k+1)$, via equation 3.11. As might be expected, by adding the

[3]This event can be detected by projecting the predicted ground plane position of the bottom-most crop plant into image co-ordinates (using the observation equations) and checking whether this prediction lies within the bounds of the image.

uncertain value of $\bar{l}(k')$, the variance on the Y prediction will increase, and also the covariances between Y and the other state variables. This is illustrated by taking the variance prediction equation

$$\mathbf{P}^-(k'+1) = \mathbf{A}(k)\mathbf{P}(k)\mathbf{A}^T(k) + \mathbf{Q}(k), \tag{4.9}$$

from which, if we drop the time indices for brevity and substitute appropriate values for $\mathbf{A}$ and $\mathbf{P}$ we see:

$$\mathbf{P}^- = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} P^2_{t_x t_x} & P^2_{t_x Y} & P^2_{t_x \Psi} & P^2_{t_x \bar{r}} & P^2_{t_x \bar{l}} \\ P^2_{t_x Y} & P^2_{YY} & P^2_{Y\Psi} & P^2_{Y\bar{r}} & P^2_{Y\bar{l}} \\ P^2_{t_x \Psi} & P^2_{Y\Psi} & P^2_{\Psi\Psi} & P^2_{\Psi\bar{r}} & P^2_{\Psi\bar{l}} \\ P^2_{t_x \bar{r}} & P^2_{Y\bar{r}} & P^2_{\Psi\bar{r}} & P^2_{\bar{r}\bar{r}} & P^2_{\bar{r}\bar{l}} \\ P^2_{t_x \bar{l}} & P^2_{Y\bar{l}} & P^2_{\Psi\bar{l}} & P^2_{\bar{r}\bar{l}} & P^2_{\bar{l}\bar{l}} \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 \end{bmatrix} + \mathbf{Q}. \tag{4.10}$$

P^2_{ab} denotes the prior covariance between state variables a and b. If we resolve the matrix product, it yields the following expression for the state prediction covariance:

$$\mathbf{P}^- = \begin{bmatrix} P^2_{t_x t_x} & P^2_{t_x Y} + P^2_{t_x \bar{l}} & P^2_{t_x \Psi} & P^2_{t_x \bar{r}} & P^2_{t_x \bar{l}} \\ P^2_{t_x Y} + P^2_{t_x \bar{l}} & P^2_{YY} + 2P^2_{Y\bar{l}} + P^2_{\bar{l}\bar{l}} & P^2_{Y\Psi} + P^2_{\Psi\bar{l}} & P^2_{Y\bar{r}} + P^2_{\bar{r}\bar{l}} & P^2_{Y\bar{l}} + P^2_{\bar{l}\bar{l}} \\ P^2_{t_x \Psi} & P^2_{Y\Psi} + P^2_{\Psi\bar{l}} & P^2_{\Psi\Psi} & P^2_{\Psi\bar{r}} & P^2_{\Psi\bar{l}} \\ P^2_{t_x \bar{r}} & P^2_{Y\bar{r}} + P^2_{\bar{r}\bar{l}} & P^2_{\Psi\bar{r}} & P^2_{\bar{r}\bar{r}} & P^2_{\bar{r}\bar{l}} \\ P^2_{t_x \bar{l}} & P^2_{Y\bar{l}} + P^2_{\bar{l}\bar{l}} & P^2_{\Psi\bar{l}} & P^2_{\bar{r}\bar{l}} & P^2_{\bar{l}\bar{l}} \end{bmatrix} + \mathbf{Q}. \tag{4.11}$$

Thus, the elements in the row and column corresponding to the Y estimate have all increased, and it can be seen that using the latest estimate for $\bar{l}$ to change the point of reference for the estimate of Y increases the uncertainty on the Y estimate and increases its covariance with the other state variables.

After time step k' and until the time at which the new bottom-most line of plants moves out of view, the $\mathbf{A}$ matrix reverts to the 5×5 identity matrix as in equation 4.4. During this period the estimate uncertainty evolves in a straightforward manner with the covariances unaffected by the state evolution matrix.

4.2 Observation model

In chapter 2, equations were given for the image positions of plant (m, n) in the crop grid, as a function of $(t_x, \Psi, Y, \bar{r}, \bar{l}, m, n)$. These may be re-written as a vector function $\mathbf{h}$, given by:

$$\mathbf{h}(t_x, \Psi, Y, \bar{r}, \bar{l}, m, n) = \begin{bmatrix} \frac{f}{dx} \frac{n\bar{r} + \Psi(m\bar{l} + Y) + t_x}{n\bar{r}\Psi \sin\phi - (m\bar{l} + Y)\sin\phi + t_z} + C_x \\ \frac{f}{dy} \frac{(m\bar{l} + Y - \Psi n\bar{r})\cos\phi}{n\bar{r}\Psi \sin\phi - (m\bar{l} + Y)\sin\phi + t_z} + C_y \end{bmatrix}. \tag{4.12}$$

The first row of this vector can be recognised as the equation for x_f (equation 2.5), the image x co-ordinate of the position of the plant at grid position (m, n), and the second line is y_f (equation 2.6), the corresponding image y co-ordinate. C_x and C_y give the image position of the camera's optic axis, whilst dx and dy define the pixel dimensions and f the focal length of the lens. With the introduction of $\mathbf{h}$ as in equation 4.12, the observed position $\mathbf{z}(k, m, n)$ of plant (m, n) at time step (k) is given by the observation model

$$\mathbf{z}(k, m, n) = \mathbf{h}(t_x, \Psi, Y, \bar{r}, \bar{l}, m, n) + \mathbf{w}(k) \tag{4.13}$$

where $\mathbf{w}(k)$ is zero mean Gaussian noise with covariance matrix $\mathbf{R}$ representing the uncertainty in measured feature position. As discussed in section 2.3.8, the noise on the measured feature position is not Gaussian, as it will be constituted of non-linear projection errors and the quantisation of the pixel sampling of the image. However, the noise *is* likely to be uni-modal, and the normal distribution is suited to the extended Kalman filter framework, so such an approximation is practically expedient. A method for determining $\mathbf{R}$ is given in section 4.2.5 below.

Equation 4.13 may be written as a function of the state vector $\mathbf{x}(k)$ simply by substituting the appropriate set of variables $[t_x, Y, \Psi]^T$ or $[t_x, Y, \Psi, \bar{r}, \bar{l}]^T$ for $\mathbf{x}(k)$ as appropriate. For the fixed parameter case, this gives

$$\mathbf{h}(\mathbf{x}(k), \bar{r}, \bar{l}, m, n) = \begin{bmatrix} x_f(t_x, Y, \Psi, \bar{r}, \bar{l}, m, n) \\ y_f(t_x, Y, \Psi, \bar{r}, \bar{l}, m, n) \end{bmatrix}, \tag{4.14}$$

and when $\bar{r}$ and $\bar{l}$ are estimated

$$\mathbf{h}(\mathbf{x}(k), m, n) = \begin{bmatrix} x_f(t_x, Y, \Psi, \bar{r}, \bar{l}, m, n) \\ y_f(t_x, Y, \Psi, \bar{r}, \bar{l}, m, n) \end{bmatrix}. \tag{4.15}$$

x_f and y_f are given in equations 2.5 and 2.6, and are also seen in equation 4.12.

4.2.1 The matrix of partial derivatives $\mathbf{h}_{\mathbf{x}(k)}$

For the extended Kalman filter computations, the matrix of partial derivatives of the observation function with respect to the state variables is required. The matrices for the two filters are given for the cases of fixed and estimated crop grid parameters, followed by the functions for the derivatives.

4.2.2 Partial derivatives matrix with fixed grid parameters

In this case, the matrix of partial derivatives is defined by:

$$\mathbf{h}_{\mathbf{x}(k)}(\mathbf{x}(k), \bar{r}, \bar{l}, m, n) = \begin{bmatrix} \left.\frac{\partial x_f}{\partial t_x}\right|_{\mathbf{x}(k),m,n} & \left.\frac{\partial x_f}{\partial Y}\right|_{\mathbf{x}(k),m,n} & \left.\frac{\partial x_f}{\partial \Psi}\right|_{\mathbf{x}(k),m,n} \\ \left.\frac{\partial y_f}{\partial t_x}\right|_{\mathbf{x}(k),m,n} & \left.\frac{\partial y_f}{\partial Y}\right|_{\mathbf{x}(k),m,n} & \left.\frac{\partial y_f}{\partial \Psi}\right|_{\mathbf{x}(k),m,n} \end{bmatrix}. \tag{4.16}$$

4.2.3 Partial derivatives matrix with estimated grid parameters

When the grid parameters are also estimated, the derivatives of $\mathbf{h}$ with respect to $\bar{r}$ and $\bar{l}$ are also required, leading to the matrix:

$$\mathbf{h}_{\mathbf{x}(k)}(\mathbf{x}(k), m, n) = \begin{bmatrix} \left.\frac{\partial x_f}{\partial t_x}\right|_{\mathbf{x}(k),m,n} & \left.\frac{\partial x_f}{\partial Y}\right|_{\mathbf{x}(k),m,n} & \left.\frac{\partial x_f}{\partial \Psi}\right|_{\mathbf{x}(k),m,n} & \left.\frac{\partial x_f}{\partial \bar{r}}\right|_{\mathbf{x}(k),m,n} & \left.\frac{\partial x_f}{\partial \bar{l}}\right|_{\mathbf{x}(k),m,n} \\ \left.\frac{\partial y_f}{\partial t_x}\right|_{\mathbf{x}(k),m,n} & \left.\frac{\partial y_f}{\partial Y}\right|_{\mathbf{x}(k),m,n} & \left.\frac{\partial y_f}{\partial \Psi}\right|_{\mathbf{x}(k),m,n} & \left.\frac{\partial y_f}{\partial \bar{r}}\right|_{\mathbf{x}(k),m,n} & \left.\frac{\partial y_f}{\partial \bar{l}}\right|_{\mathbf{x}(k),m,n} \end{bmatrix} \quad (4.17)$$

4.2.4 The partial derivatives

The partial derivatives appearing in equations 4.16 and 4.17 may be obtained from equations 2.5 and 2.6 by differentiation, resulting in:

$$\frac{\partial x_f}{\partial t_x} = \frac{f}{dx}\frac{1}{((\Psi n\bar{r} - m\bar{l} - Y)\sin\phi + t_z)}, \quad (4.18)$$

$$\frac{\partial x_f}{\partial Y} = \frac{f}{dx}\frac{((\Psi n\bar{r} - m\bar{l} - Y)\sin\phi + t_z)\Psi + (n\bar{r} + \Psi(m\bar{l} + Y) + t_x)\sin\phi}{((\Psi n\bar{r} - m\bar{l} - Y)\sin\phi + t_z)^2}, \quad (4.19)$$

$$\frac{\partial x_f}{\partial \Psi} = \frac{f}{dx}\frac{((\Psi n\bar{r} - m\bar{l} - Y)\sin\phi + t_z)(m\bar{l} + Y) - (n\bar{r} + \Psi(m\bar{l} + Y) + t_x)n\bar{r}\sin\phi}{((\Psi n\bar{r} - m\bar{l} - Y)\sin\phi + t_z)^2}, \quad (4.20)$$

$$\frac{\partial x_f}{\partial \bar{r}} = \frac{f}{dx}\frac{((\Psi n\bar{r} - m\bar{l} - Y)\sin\phi + t_z)n - (n\bar{r} + \Psi(m\bar{l} + Y) + t_x)n\Psi\sin\phi}{((\Psi n\bar{r} - m\bar{l} - Y)\sin\phi + t_z)^2}, \quad (4.21)$$

$$\frac{\partial x_f}{\partial \bar{l}} = \frac{f}{dx}\frac{((\Psi n\bar{r} - m\bar{l} - Y)\sin\phi + t_z)m\Psi + (n\bar{r} + \Psi(m\bar{l} + Y) + t_x)m\sin\phi}{((\Psi n\bar{r} - m\bar{l} - Y)\sin\phi + t_z)^2}, \quad (4.22)$$

and

$$\frac{\partial y_f}{\partial t_x} = 0, \quad (4.23)$$

$$\frac{\partial y_f}{\partial Y} = \frac{f}{dy}\frac{t_z\cos\phi}{((\Psi n\bar{r} - m\bar{l} - Y)\sin\phi + t_z)^2}, \quad (4.24)$$

$$\frac{\partial y_f}{\partial \Psi} = \frac{-f}{dy}\frac{t_z n\bar{r}\cos\phi}{((\Psi n\bar{r} - m\bar{l} - Y)\sin\phi + t_z)^2}, \quad (4.25)$$

$$\frac{\partial y_f}{\partial \bar{r}} = \frac{-f}{dy}\frac{t_z n\Psi\cos\phi}{((\Psi n\bar{r} - m\bar{l} - Y)\sin\phi + t_z)^2}, \quad (4.26)$$

$$\frac{\partial y_f}{\partial \bar{l}} = \frac{f}{dy}\frac{t_z m\cos\phi}{((\Psi n\bar{r} - m\bar{l} - Y)\sin\phi + t_z)^2}. \quad (4.27)$$

All the symbols used are defined as above and in chapter 2.

4.2.5 Observation noise

The zero-mean Gaussian observation noise is quantified by a covariance matrix $\mathbf{R}$. As we noted above, the noise is unlikely to be truly Gaussian, but the assumption is practically expedient. Observation noise reflects the uncertainty in the measurement process, and in the crop grid tracking application the measurement process is image processing. The measurements taken from the image are the centroids of blob features extracted from the image, and the assumption is that a blob centroid represents the position of a plant (more specifically, its root) in the crop grid.

From the discussion of section 2.3.8, we know that the blob centroid is unlikely to be an accurate reflection of the plant position, but that the true centre is likely to lie within or near the feature. The observation noise covariance $\mathbf{R}$ must reflect this proximity, which is most easily expressed on the ground plane. We can project the feature centroid, at pixel co-ordinate x_{fp}, y_{fp}, onto the ground plane point x_{wp}, y_{wp} by using the camera calibration. The uncertainty of feature position on the ground plane may be expressed by the covariance matrix $\mathbf{R}_w = \text{diag}[\sigma^2_{x_w}, \sigma^2_{y_w}]$, centred on x_{wp}, y_{wp}. The terms $\sigma^2_{x_w}$ and $\sigma^2_{y_w}$ describe the variance of the observation noise on the ground plane and are set empirically. The observation covariance matrix $\mathbf{R}$ should express the noise variance in terms of pixels, so we need to project the matrix $\mathbf{R}_w$ into the image, at pixel x_{fp}, y_{fp}. This is done using standard first order error propagation techniques like so,

$$\mathbf{R}(x_{fp}, y_{fp}) = \mathbf{F}_w(x_{wp}, y_{wp})\mathbf{R_w}\mathbf{F}^T_w(x_{wp}, y_{wp}). \tag{4.28}$$

Where the matrix of partial derivatives $\mathbf{F}_w(x_{wp}, y_{wp})$, that performs the projection, is defined by

$$\mathbf{F}_w(x_{wp}, y_{wp}) = \begin{bmatrix} \left.\frac{\partial x_f}{\partial x_w}\right|_{x_{wp},y_{wp}} & \left.\frac{\partial x_f}{\partial y_w}\right|_{x_{wp},y_{wp}} \\ \left.\frac{\partial y_f}{\partial x_w}\right|_{x_{wp},y_{wp}} & \left.\frac{\partial y_f}{\partial x_w}\right|_{x_{wp},y_{wp}} \end{bmatrix}. \tag{4.29}$$

In equation 4.29, partial derivatives of the function describing the image pixel co-ordinate x_f and y_f are required in terms of the world ground plant co-ordinate x_w and y_w. The functions x_f and y_f, derived from equations 2.3 and 2.4 in chapter 2 are given below:

$$x_f = \frac{f}{dx}\frac{(x_w \cos\Psi + y_w \sin\Psi + t_x)}{x_w \sin\Psi \sin\phi - y_w \cos\Psi \sin\phi + t_z} + C_x, \tag{4.30}$$

$$y_f = \frac{f}{dy}\frac{(-x_w \sin\Psi \cos\phi + y_w \cos\Psi \cos\phi + t_y)}{x_w \sin\Psi \sin\phi - y_w \cos\Psi \sin\phi + t_z} + C_y, \tag{4.31}$$

where Ψ and t_x come from the vehicle's state estimate, ϕ is the angle of the camera's optic axis to the vertical, f is the focal length of the camera's lens, dx and dy the dimensions of the pixels on the image plane and C_x, C_y the pixel co-ordinate of the camera's optic axis.

4.3 Alternative filter formulations

There are many alternative implementations of the Kalman filter and extended Kalman filter algorithms presented in the previous chapter, which have differing computational and explanatory benefits. Three such implementations are detailed in this thesis, starting with the information filter, also known as the inverse covariance filter. This formulation is not of direct concern to the filter systems implemented, but is given to aid explanation of the parallel update and data compression filters detailed subsequently, and to show equivalence between the alternative filters and the standard Kalman filter algorithms as given in chapter 3.

The second alternative formulation given is the parallel update filter, where a batch of observations are integrated into the state estimate simultaneously. This implementation will prove useful in demonstrating the observability properties of the two off-line filters detailed above. At the end of this chapter, the on-line vision system is described; to aid this description the third Kalman filter variant, the data compression filter, is developed in section 4.6.

4.3.1 The information filter

The information filter, or inverse covariance filter [May79], is a formulation of the Kalman filter which has update rules for the inverse of the state covariance matrix, $\mathbf{P}^{-1}(k)$, and the product of this inverse matrix and the state estimate, $\mathbf{P}^{-1}(k)\hat{\mathbf{x}}(k)$. In practice, the information filter may offer increased numerical stability over the standard filter equations, in particular when the state covariance matrix has small condition number. In such situations the inverse covariance matrix will be better conditioned. However, such poor conditioning is not a problem in our filters[4], and the information filter form is used in this thesis solely to aid explanation of the parallel update and data compression (section 4.6) filters.

The update equations of the linear information filter are

$$\mathbf{P}^{-1}(k+1) = (\mathbf{P}^{-})^{-1}(k+1) + \mathbf{H}^{T}(k+1)\mathbf{R}^{-1}(k+1)\mathbf{H}(k+1), \tag{4.32}$$

and

$$\mathbf{P}^{-1}(k+1)\hat{\mathbf{x}}(k+1) = (\mathbf{P}^{-})^{-1}(k+1)\hat{\mathbf{x}}^{-}(k+1) + \mathbf{H}^{T}(k+1)\mathbf{R}^{-1}(k+1)\mathbf{z}(k+1). \tag{4.33}$$

Equation 4.32 is clearly the inverse of the Kalman filter covariance derived in equation 3.24, whilst the state update equation (equation 4.33) may be derived from the standard Kalman filter update equation

$$\hat{\mathbf{x}}(k) = \hat{\mathbf{x}}^{-}(k) + \mathbf{P}^{-}(k)\mathbf{H}^{T}(k)[\mathbf{H}(k)\mathbf{P}^{-}(k)\mathbf{H}^{T}(k) + \mathbf{R}(k)]^{-1}(\mathbf{z}(k) - \mathbf{H}(k)\hat{\mathbf{x}}^{-}(k)), \tag{4.34}$$

[4] We maintain estimates of t_x,Y,$\bar{r}$ and $\bar{l}$ in metres and Ψ in radians to ensure that the elements of the covariance matrices are of similar magnitude.

as follows. First, multiply both sides of equation 4.34 by $\mathbf{P}^{-1}$ (the time indices have been dropped for brevity):

$$\mathbf{P}^{-1}\hat{\mathbf{x}} = \mathbf{P}^{-1}[\hat{\mathbf{x}}^- + \mathbf{P}^-\mathbf{H}^T[\mathbf{H}\mathbf{P}^-\mathbf{H}^T + \mathbf{R}]^{-1}(\mathbf{z} - \mathbf{H}\hat{\mathbf{x}}^-)], \tag{4.35}$$

and substitute equation 4.32 into the right hand side:

$$\mathbf{P}^{-1}\hat{\mathbf{x}} = [(\mathbf{P}^-)^{-1} + \mathbf{H}^T\mathbf{R}^{-1}\mathbf{H}][\hat{\mathbf{x}}^- + \mathbf{P}^-\mathbf{H}^T[\mathbf{H}\mathbf{P}^-\mathbf{H}^T + \mathbf{R}]^{-1}(\mathbf{z} - \mathbf{H}\hat{\mathbf{x}}^-)]. \tag{4.36}$$

If we then re-arrange

$$\begin{aligned}\mathbf{P}^{-1}\hat{\mathbf{x}} = {} & [(\mathbf{P}^-)^{-1} + \mathbf{H}^T\mathbf{R}^{-1}\mathbf{H}]\hat{\mathbf{x}}^- + \\ & [\mathbf{H}^T[\mathbf{H}(\mathbf{P}^-)^{-1}\mathbf{H}^T + \mathbf{R}]^{-1}] + \mathbf{H}^T\mathbf{R}^{-1}\mathbf{H}\mathbf{P}^-\mathbf{H}^T[\mathbf{H}(\mathbf{P}^-)^{-1}\mathbf{H}^T + \mathbf{R}]^{-1}[\mathbf{z} - \mathbf{H}\hat{\mathbf{x}}^-],\end{aligned} \tag{4.37}$$

factorise,

$$\begin{aligned}\mathbf{P}^{-1}\hat{\mathbf{x}} = {} & [(\mathbf{P}^-)^{-1} + \mathbf{H}^T\mathbf{R}^{-1}\mathbf{H}]\hat{\mathbf{x}}^- + \\ & \mathbf{H}^T\mathbf{R}^{-1}[\mathbf{R} + \mathbf{H}(\mathbf{P}^-)^{-1}\mathbf{H}^T][\mathbf{R} + \mathbf{H}(\mathbf{P}^-)^{-1}\mathbf{H}^T]^{-1}[\mathbf{z} - \mathbf{H}\hat{\mathbf{x}}^-],\end{aligned} \tag{4.38}$$

and finally simplify (and reinstate the time step), we produce equation 4.33 above

$$\mathbf{P}^{-1}(k+1)\hat{\mathbf{x}}(k+1) = (\mathbf{P}^-)^{-1}(k+1)\hat{\mathbf{x}}^-(k+1) + \mathbf{H}^T(k+1)\mathbf{R}^{-1}(k+1)\mathbf{z}(k+1). \tag{4.39}$$

In filters with non-linear observation function ($\mathbf{z}(k) = \mathbf{h}(\mathbf{x}(k))$) in which the matrix of partial derivatives of $\mathbf{h}(\mathbf{x}(k))$ evaluated at the prediction point is denoted $\mathbf{h}_{\hat{\mathbf{x}}^-(k)}$, the information filter equations 4.32 and 4.33 become

$$\mathbf{P}^{-1}(k+1) = (\mathbf{P}^-)^{-1}(k+1) + \mathbf{h}^T_{\hat{\mathbf{x}}^-(k+1)}\mathbf{R}^{-1}(k+1)\mathbf{h}_{\hat{\mathbf{x}}^-(k+1)}, \tag{4.40}$$

and

$$\begin{aligned}\mathbf{P}^{-1}(k+1)\hat{\mathbf{x}}(k+1) = {} & [(\mathbf{P}^-)^{-1}(k+1) + \mathbf{h}^T_{\hat{\mathbf{x}}^-(k+1)}(k+1)\mathbf{R}^{-1}(k+1)\mathbf{h}_{\hat{\mathbf{x}}^-(k+1)}]\hat{\mathbf{x}}^-(k+1) + \\ & \mathbf{h}^T_{\hat{\mathbf{x}}^-(k+1)}(k+1)\mathbf{R}^{-1}(k+1)[\mathbf{z}(k+1) - \mathbf{h}(\hat{\mathbf{x}}^-(k+1))],\end{aligned} \tag{4.41}$$

respectively. With the equivalence between the information filter and standard Kalman filter of chapter 3 demonstrated, the parallel update filter can be introduced and its equivalence with the Kalman filter proved in a straightforward manner.

4.3.2 The parallel update filter

The Kalman filter mechanism described in chapter 3 is essentially a serial update filter. At each time-step k, a single observation is incorporated into the filter. When several measurements become available simultaneously[5], an equivalent method may be used which incorporates all of the simultaneous observations in a single step, i.e. a parallel update Kalman filter

[5]For example, when many features are extracted from a single image.

[WCD76, LDW91a]. The basic method is to construct batch quantities containing information about the observations, which are then used in the standard Kalman filter equations as explained below. The equivalence between the serial and parallel update methods is also shown.

If $\mathbf{z}(k+1,s)$ is the s^{th} measurement of S collected at time $k+1$ (one measurement s for each m,n grid index pair), with observation matrix $\mathbf{H}(k+1,s)$, then the batch quantities describing the set of observations and their observation matrices may be formed by stacking the S measurements and matrices as follows:

$$\mathbf{z}(k+1) = \begin{bmatrix} \mathbf{z}(k+1,1) \\ \mathbf{z}(k+1,2) \\ \vdots \\ \mathbf{z}(k+1,S) \end{bmatrix} \tag{4.42}$$

$$\mathbf{H}(k+1) = \begin{bmatrix} \mathbf{H}(k+1,1) \\ \mathbf{H}(k+1,2) \\ \vdots \\ \mathbf{H}(k+1,S) \end{bmatrix}. \tag{4.43}$$

An associated measurement covariance matrix $\mathbf{R}(k+1)$ is composed of the S covariances $\mathbf{R}(k+1,s)$

$$\mathbf{R}(k+1) = \begin{bmatrix} \mathbf{R}(k+1,1) & 0 & \dots & 0 \\ 0 & \mathbf{R}(k+1,2) & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & \mathbf{R}(k+1,S) \end{bmatrix}. \tag{4.44}$$

These three block quantities may now be substituted directly into the filter update equations (equations 3.12, 3.13 and 3.14). Again, for the non-linear case, the matrix $\mathbf{H}$ is simply replaced by the matrix of partial derivatives of the observation function with respect to the state variables.

4.3.3 Equivalence of update schemes

The information filter was introduced above in order to demonstrate simply the equivalence between the parallel update filter and the standard recursive formulation. To show the equivalence, we begin with the update equations for the information filter:

$$\mathbf{P}^{-1}(k+1) = (\mathbf{P}^{-})^{-1}(k+1) + \mathbf{H}^{T}(k+1)\mathbf{R}^{-1}(k+1)\mathbf{H}(k+1), \tag{4.45}$$

$$\mathbf{P}^{-1}(k+1)\hat{\mathbf{x}}(k+1) = (\mathbf{P}^{-})^{-1}(k+1)\hat{\mathbf{x}}^{-}(k+1) + \mathbf{H}^{T}(k+1)\mathbf{R}^{-1}(k+1)\mathbf{z}(k+1). \tag{4.46}$$

Given a set of S measurements $\mathbf{z}(k+1,s)$ and corresponding matrices $\mathbf{H}(k+1,s)$ and $\mathbf{R}(k+1,s)$, the inverse covariance and projected state estimate after the incorporation of the S mea-

surements will be

$$\mathbf{P}^{-1}(k{+}1)_S = (\mathbf{P}^-)^{-1}(k{+}1)_0 + \sum_{s=1}^{S} \mathbf{H}^T(k{+}1,s)\mathbf{R}^{-1}(k{+}1,s)\mathbf{H}(k{+}1,s), \tag{4.47}$$

$$\mathbf{P}^{-1}(k{+}1)\hat{\mathbf{x}}(k{+}1)_S = (\mathbf{P}^-)^{-1}(k{+}1)\hat{\mathbf{x}}^-(k{+}1)_0 + \sum_{s=1}^{S} \mathbf{H}^T(k{+}1,s)\mathbf{R}^{-1}(k{+}1,s)\mathbf{z}(k{+}1,s), \tag{4.48}$$

where the subscripts denote the number of incorporated observations. The summation terms in equations 4.47 and 4.48 are a consequence of the form of the information filter update equations 4.45 and 4.46, where new information is incorporated by simple addition of terms relating to the observation and its covariance[6]. To show the equivalence of this serial update method with the parallel update method, all that is required is to notice that the summation terms in equations 4.47 and 4.48 may be more concisely written as matrix multiplications

$$\mathbf{P}^{-1}(k{+}1)_S = (\mathbf{P}^-)^{-1}(k{+}1)_0 + \mathbf{H}^T_{stack}(k{+}1)\mathbf{R}^{-1}_{stack}(k{+}1)\mathbf{H}_{stack}(k{+}1), \tag{4.49}$$

$$\mathbf{P}^{-1}(k{+}1)\hat{\mathbf{x}}(k{+}1)_S = (\mathbf{P}^-)^{-1}(k{+}1)\hat{\mathbf{x}}^-(k{+}1)_0 + \mathbf{H}^T_{stack}(k{+}1)\mathbf{R}^{-1}_{stack}(k{+}1)\mathbf{z}_{stack}(k{+}1). \tag{4.50}$$

The matrices $\mathbf{H}_{stack}(k+1)$, $\mathbf{R}^{-1}_{stack}(k+1)$ and $\mathbf{z}_{stack}(k+1)$ are composed in the following way

$$\mathbf{H}_{stack}(k+1) = \begin{bmatrix} \mathbf{H}(k+1,1) \\ \mathbf{H}(k+1,2) \\ \vdots \\ \mathbf{H}(k+1,S) \end{bmatrix}, \tag{4.51}$$

$$\mathbf{R}^{-1}_{stack}(k+1) = \begin{bmatrix} \mathbf{R}^{-1}(k+1,1) & 0 & \dots & 0 \\ 0 & \mathbf{R}^{-1}(k+1,2) & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & \mathbf{R}^{-1}(k+1,S) \end{bmatrix}, \tag{4.52}$$

$$\mathbf{z}_{stack}(k+1) = \begin{bmatrix} \mathbf{z}(k+1,1) \\ \mathbf{z}(k+1,2) \\ \vdots \\ \mathbf{z}(k+1,S) \end{bmatrix}. \tag{4.53}$$

Equations 4.53 and 4.51 are familiar from the parallel Kalman filter equations 4.42 and 4.43, and 4.52 is simply the inverse of the matrix given by equation 4.44. Hence, the parallel update

[6]This highlights another contrast with the Kalman filter equations of chapter 3. In the information filter, state update has complexity $O(n^2)$ (where n is the dimension of the state vector), and prediction $O(n^3)$. When the Kalman filter is implemented with the equations of chapter 3, prediction is simpler at $O(n^2)$ and the update step more complex at $O(n^3)$.

and serial update Kalman filters are equivalent. Again, in non-linear systems, a matrix of partial derivatives $\mathbf{h}_{\hat{\mathbf{x}}(k)_{stack}}$ replaces $\mathbf{H}_{stack}$:

$$\mathbf{h}_{\hat{\mathbf{x}}(k)_{stack}} = \begin{bmatrix} \mathbf{h}_{\hat{\mathbf{x}}(k+1,1)} \\ \mathbf{h}_{\hat{\mathbf{x}}(k+1,2)} \\ \vdots \\ \mathbf{h}_{\hat{\mathbf{x}}(k+1,S)} \end{bmatrix}. \tag{4.54}$$

An implementation of the non-linear parallel update filter is used in the off-line experiments detailed in section 5.1 of the next chapter.

4.4 Corruptibility of the crop grid tracker

We now return to study the off-line filters whose process and observation models were defined in sections 4.1 and 4.2. Given the state evolution models described in section 4.1, the corruptibility of the $\mathbf{A}, \mathbf{V}$ pair of the two systems may be assessed using the method defined previously in section 3.4.5. It will be shown that, on the one hand, the filter with fixed grid parameters has a completely corruptible $\mathbf{A}, \mathbf{V}$ pair, whilst on the other hand, the filter with estimated grid parameters is only partially corruptible.

4.4.1 Corruptibility with fixed grid parameters

The state evolution model for the filter with fixed grid parameters is, repeated here from equation 4.1,

$$\mathbf{x}(k+1) = \mathbf{I}_3\mathbf{x}(k) + \mathbf{I}_3 \begin{bmatrix} U_{t_x} \\ U_Y \\ U_\Psi \end{bmatrix} + \mathbf{V}\mathbf{v}(k), \tag{4.55}$$

where $\mathbf{I}_3$ is the 3×3 identity matrix, and the subscripted U values describe vehicle motion between image acquisitions. The noise gain matrix is given by

$$\mathbf{V}^T(k)\mathbf{V}(k) = \mathbf{Q}(k) = \begin{bmatrix} \sigma^2_{t_x}(k) & 0 & 0 \\ 0 & \sigma^2_Y(k) & 0 \\ 0 & 0 & \sigma^2_\Psi(k) \end{bmatrix}. \tag{4.56}$$

It is immediately clear, from the fact that in equation 4.55 the matrix $\mathbf{A}$ is the identity matrix, that, provided none of the diagonal terms of $\mathbf{V}$ collapse to zero, the $\mathbf{A}, \mathbf{V}$ pair form a corruptible system as discussed in section 3.4.4. The noise inputs $\mathbf{v}(k)$ will thus always filter through to the state estimate.

4.4.2 Corruptibility with estimated grid parameters

The second filter described in section 4.1 estimates the values of the grid parameters $\bar{r}$ and $\bar{l}$ in addition to estimating the vehicle position. Equations 4.4 and 4.5, which specify the filter's state evolution model and noise gain matrix respectively, are repeated below for clarity.

$$\mathbf{x}(k+1) = \mathbf{A}(k)\mathbf{x}(k) + \mathbf{I}_5 \begin{bmatrix} U_{t_x} \\ U_Y \\ U_\Psi \\ 0 \\ 0 \end{bmatrix} + \mathbf{V}\mathbf{v}(k). \tag{4.57}$$

In this equation, depending on whether the line of plants used as a reference for the Y parameter is undergoing change or not, $\mathbf{A}(k)$ is either $\mathbf{I}_5$, the 5×5 identity matrix when no change occurs, or is the matrix given in equation 4.8 when the reference point does switch. The subscripted U values are control inputs relating the vehicle's motion between image acquisitions. It should be noted that, except for noise, the mean plant spacings $\bar{r}$ and $\bar{l}$ are assumed constant. The noise gain matrix is diagonal, such that

$$\mathbf{V}^T(k)\mathbf{V}(k) = \mathbf{Q}(k) = \begin{bmatrix} \sigma_{t_x}^2(k) & 0 & 0 & 0 & 0 \\ 0 & \sigma_Y^2(k) & 0 & 0 & 0 \\ 0 & 0 & \sigma_\Psi^2(k) & 0 & 0 \\ 0 & 0 & 0 & \sigma_{\bar{r}}^2(k) & 0 \\ 0 & 0 & 0 & 0 & \sigma_{\bar{l}}^2(k) \end{bmatrix}. \tag{4.58}$$

The σ^2 terms relate to the mean-square error introduced at each prediction step. For t_x, Y and Ψ they are relate to the uncertainty on the control inputs, but for $\bar{r}$ and $\bar{l}$ the σ^2 terms are non-zero *only* when the set of plants in the image changes, i.e. when a new set of plants appear in the image, or when previously seen plants move out of the field of view. The reason for this is that the system noise $\mathbf{Q}$ represents the uncertainty in the evolution of the state variables, and, because they describe the mean grid parameters for the set of plants in the image, $\bar{r}$ and $\bar{l}$ evolve *only* when new plants come into the field of view or previously seen plants are passed by and no longer seen. By inspection of $\mathbf{Q}(k)$ with $\sigma_{\bar{r}}^2$ and $\sigma_{\bar{l}}^2$ zero (i.e. when no new plants have come into or moved out of view) it can be seen that the $\mathbf{AV}$ pair is incorruptible regardless of the value of $\mathbf{A}(k)$, owing to the zeroes in $\mathbf{V}$ ($\mathbf{V} = \sqrt{\mathbf{Q}(k)} = \mathrm{diag}(\sigma_{t_x}, \sigma_Y, \sigma_\Psi, 0, 0)$). When this is the case, the estimated covariance on the estimates of both $\bar{r}$ and $\bar{l}$ will start to decrease. This is quite acceptable, however, because during the period where the set of plants in view does not change, the filter is refining its estimate of the plant spacing using several views of the same patch of

crop. However, when new plants are seen or plants go out of view, process noise *is* added which reflects the fact that the mean grid spacing parameters may well be different for the crop in the new scene. In short, when the mean plant spacing parameters are estimated, the filter should be fully corruptible only some of the time, and we shall call this partial incorruptibility.

To illustrate corruptibility and partial incorruptibility, a fictitious system was devised whose process is identical to that described by equation 4.57 in every way except that noise is added to the $\bar{r}$ and $\bar{l}$ with each new image, so it is fully corruptible. Figure 4.2 shows the estimate of $\bar{r}$ over a set of 20 images for the partially incorruptible system described above (figure 4.2, right) and the illustrative system, where noise is added to $\bar{r}$ with each new image (figure 4.2, left). The partially incorruptible system shows an increase in variance at frames 4 and 12 which corresponds to noise added as the set of crop plants in the scene changes. Between these noise injections, the filter estimate starts to converge. The estimate from the corruptible system is not as smooth as that from the incorruptible one, and the variance of the estimate does not shrink over the sequence. Both of these behaviours are to be expected when new plant noise is injected for each image. Similar behaviour has been observed for the estimate of $\bar{l}$.

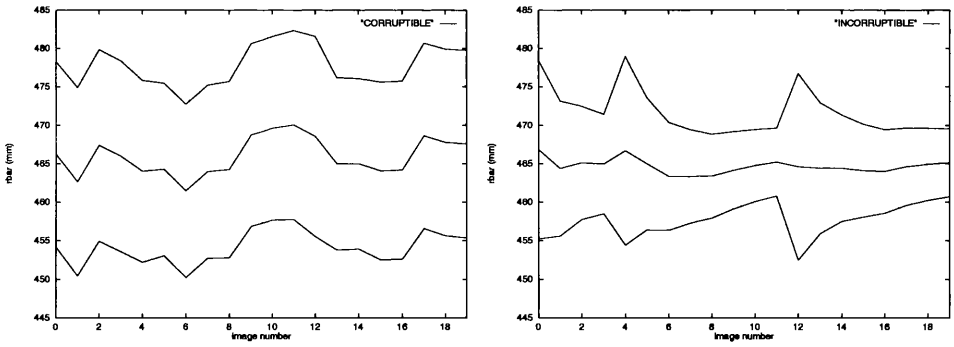


Figure 4.2: A comparison of state estimates and variances from a corruptible and partially incorruptible filter. The middle trace shows the state estimate, the upper and lower traces showing the estimate ± 2 standard deviations. Left: corruptible estimates. Right: partially incorruptible system estimates. See text for details.

4.5 Observability of the crop grid tracker

In section 3.5, a criterion for the observability of a system with linear evolution model and non-linear measurement model was derived, namely that the row rank of the matrix

$$\mathcal{O}_{lin} = \begin{bmatrix} \mathbf{h}_{\hat{\mathbf{x}}(0)} \\ \mathbf{h}_{\hat{\mathbf{x}}(1)}\mathbf{A} \\ \vdots \\ \mathbf{h}_{\hat{\mathbf{x}}(i)}\mathbf{A}^i \\ \vdots \\ \mathbf{h}_{\hat{\mathbf{x}}(n-1)}\mathbf{A}^{n-1} \end{bmatrix}, \tag{4.59}$$

should be n, where n is the dimension of the state vector $\mathbf{x}$. It was also pointed out that this condition will be met if the matrix of partial derivatives $\mathbf{h}_{\hat{\mathbf{x}}(k)}$ has rank n regardless of the value of the state estimate $\hat{\mathbf{x}}(k)$. If this is not the case, then the observability may well depend on the state estimate and is not guaranteed. This will, of course, depend on the exact form of $\mathbf{h}_{\hat{\mathbf{x}}(k)}$.

In the case of the parallel update crop grid tracker, the matrix $\mathbf{h}_{\hat{\mathbf{x}}(k)}$ of interest is the stacked matrix $\mathbf{h}_{\hat{\mathbf{x}}(k)_{stack}}$ (equation 4.54), which must have a minimum of three linearly independent rows (one for each component t_x, Y and Ψ of the grid tracker's state vector) for the filter that estimates the crop grid position alone, or five linearly independent rows in the case of the filter which also estimates the crop grid parameters $\bar{r}$ and $\bar{l}$. We shall look at each case below.

4.5.1 Observability with fixed parameters

The matrix $\mathbf{h}_{\hat{\mathbf{x}}(k)_{stack}}$ is composed of the individual partial derivative matrices (equation 4.16) that correspond to each observed crop plant position. It can be seen from equation 4.54 that for the filter with fixed grid parameters, each crop plant position (m, n) contributes an associated matrix of the form

$$\mathbf{h}_{\hat{\mathbf{x}}(k)} = \mathbf{F} \begin{bmatrix} 1 & \Psi + D & m\bar{l} + Y - Dn\bar{r} \\ 0 & 1 & -n\bar{r} \end{bmatrix}, \tag{4.60}$$

where

$$\mathbf{F} = \begin{bmatrix} A(\hat{\mathbf{x}}, m, n) & 0 \\ 0 & B(\hat{\mathbf{x}}, m, n) \end{bmatrix} \tag{4.61}$$

is a matrix containing common factors $(A(\cdot), B(\cdot))$ from each row of $\mathbf{h}_{\hat{\mathbf{x}}(k)}$, and

$$D = \frac{(n\bar{r} + \Psi(m\bar{l} + Y) + t_x)\sin\phi}{(\Psi n\bar{r} - m\bar{l} - Y)\sin\phi + t_z}. \tag{4.62}$$

It should be noted that the variables t_x, Y and Ψ take their values from the elements of the state estimate $\hat{\mathbf{x}}(k)$, whilst $\bar{r}$ and $\bar{l}$ are fixed.

If there are S predicted crop grid positions m_s, n_s (see figure 4.1), with corresponding observations[7], the resulting stacked matrix (equation 4.54) is given by

$$\mathbf{h}_{\hat{\mathbf{x}}(k)_{stack}} = \mathbf{F}_{stack} \begin{bmatrix} 1 & \Psi + D & m_1\bar{l} + Y - Dn_1\bar{r} \\ 0 & 1 & -n_1\bar{r} \\ \vdots & \vdots & \vdots \\ 1 & \Psi + D & m_s\bar{l} + Y - Dn_s\bar{r} \\ 0 & 1 & -n_s\bar{r} \\ \vdots & \vdots & \vdots \\ 1 & \Psi + D & m_S\bar{l} + Y - Dn_S\bar{r} \\ 0 & 1 & -n_S\bar{r} \end{bmatrix}, \tag{4.63}$$

where the common factors from each row of the matrix are to be found in the stacked matrix $\mathbf{F}_{stack}$. The observability of the parallel update filter, and hence the equivalent serial filter of chapter 3, may now be assessed by performing a rank analysis of the stacked matrix $\mathbf{h}_{\hat{\mathbf{x}}_{stack}}$. To aid our rank analysis, we recall that each observed crop grid position m_s, n_s contributes a matrix of the form of $\mathbf{h}_{\hat{\mathbf{x}}}$ (equation 4.60) to the stacked matrix. Inspection of first row of the partial derivative matrix $\mathbf{h}_{\hat{\mathbf{x}}}$ shows that, regardless of the values of the state variables (t_x, Y, Ψ), i.e. whatever the point of linearisation of equation 4.12, the stacked matrix $\mathbf{h}_{\hat{\mathbf{x}}_{stack}}$ will have one independent row for each different m_s for which an observation is available (provided that the fixed value of $\bar{l}$ is not zero). Similarly, the second row of the partial derivative matrix will contribute an independent row to th stacked matrix for each different n_s for which an observation is available (provided that the fixed value of $\bar{r}$ is not zero). Additionally, there is no linear dependence between the two rows of $\mathbf{h}_{\hat{\mathbf{x}}}$ (equation 4.60).

To ensure observability of the parallel update filter with fixed grid parameters, we require a minimum of three linearly independent rows in the matrix $\mathbf{h}_{\hat{\mathbf{x}}_{stack}}$. A minimum of two different observations will thus suffice as is gives an $\mathbf{h}_{\hat{\mathbf{x}}_{stack}}$ with four rows. This result confirms that given by common sense; one feature can partially locate the grid structure (giving t_x and Y) and a second will determine the orientation (Ψ).

4.5.2 Observability with estimated parameters

The observability of the filter which estimates the grid parameters may be assessed using similar reasoning to that above. In this case the matrix of partial derivatives is (from equation 4.17)

$$\mathbf{h}_{\hat{\mathbf{x}}(k)} = \mathbf{F} \begin{bmatrix} 1 & \Psi + D & m\bar{l} + Y - Dn\bar{r} & n(1 - D\Psi) & m(\Psi + D) \\ 0 & 1 & -n\bar{r} & -n\Psi & m \end{bmatrix}, \tag{4.64}$$

[7] Finding observations which correspond to predictions is the process of data association, which will be addressed in chapter 6.

where $\mathbf{F}$ and D are given in equation 4.61 and 4.62 respectively. The variables $\hat{t}_x$, Y, Ψ, $\bar{r}$ and $\bar{l}$ in equation 4.64 take their values from the elements of the state estimate $\hat{\mathbf{x}}$.

The stacked observation matrix for S observations is given by

$$\mathbf{h}_{\hat{\mathbf{x}}(k)_{stack}} = \mathbf{F}_{stack} \begin{bmatrix} 1 & \Psi+D & m_1\bar{l}+Y-Dn_1\bar{r} & n_1(1-D\Psi) & m_1(\Psi+D) \\ 0 & 1 & -n_1\bar{r} & -n_1\Psi & m_1 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & \Psi+D & m_s\bar{l}+Y-Dn_s\bar{r} & n_s(1-D\Psi) & m_s(\Psi+D) \\ 0 & 1 & -n_s\bar{r} & -n_s\Psi & m_s \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & \Psi+D & m_S\bar{l}+Y-Dn_S\bar{r} & n_S(1-D)\Psi & m_S(\Psi+D) \\ 0 & 1 & -n_S\bar{r} & -n_S\Psi & m_S \end{bmatrix}, \tag{4.65}$$

where the common factors from each row of the matrix are to be found in the stacked matrix $\mathbf{F}_{stack}$. Again, we determine the observability of the system by inspection of the matrices $\mathbf{h}_{\hat{\mathbf{x}}(k)_{stack}}$ and $\mathbf{h}_{\hat{\mathbf{x}}(k)}$, which are now given by equations 4.65 and 4.64. Inspection of the second row of the partial derivative matrices of equation 4.64 shows that, regardless of the values of the state variables $(t_x, Y, \Psi, \bar{r}, \bar{l})$, i.e. whatever the point of linearisation of equation 4.12, the stacked matrix $\mathbf{h}_{\hat{\mathbf{x}}_{stack}}$ will have one independent row for each different m_s for which an observation is available. For the first row of equation 4.64 to be unique for different values of m and n places some constraints on the state variables. However, a little thought shows that sufficient conditions are that $\bar{r} \neq 0$ and $\bar{l} \neq 0$, which stipulates that the crop must lie on a two dimensional grid (i.e. that the model of figure 4.1 is valid).

When we estimate $\bar{r}$ and $\bar{l}$, five independent rows are required for observability, so a minimum of three observations are required, which give six linearly independent rows in $\mathbf{h}_{\mathbf{x}stack}$, provided that all three observations do not lie within a single row (and share a common n) or line (a common m) of the grid structure. As above, two observations will locate and orient the grid pattern and also allow estimation of the grid parameter $\bar{r}$ (or $\bar{l}$), and the third observation, provided that it does not lie on the same row (or line) of crop plants will yield the second grid parameter $\bar{l}$ (or $\bar{r}$).

4.6 The on-line vision system

The previous sections have been concerned with the development of two filters designed to run on a desktop workstation. This off-line system was used for algorithm development and testing on pre-recorded image sequences as described in chapter 5, but a little re-designing is required

to transfer the system onto the autonomous vehicle, where it operates on-line. The vehicle's on-line navigation system is comprised of two estimators. The first is an extended Kalman filter performing "dead-reckoning" by merging odometry and accelerometer data to form estimates of position, speed and acceleration at a rate of 50 Hz (section 1.2). The second estimate is the vision system filter (again, an EKF), which provides updates of position at 12.5 frames per second, to correct for any drift in the dead-reckoning system. To date, only the filter with fixed grid parameters $\bar{r}$ and $\bar{l}$ (section 4.1.1) has been implemented on-line, although, as the off-line experiments in chapter 5 will demonstrate, estimation of $\bar{r}$ and $\bar{l}$ is desirable when the crop grid spacing differs from that expected.

The different data rates of the two filters encourages a modular approach to the on-line system architecture (this was also influenced by the use of Transputers in the original hardware architecture of the vehicle), so a Kalman filter formulation that allows separation of computation whilst keeping state estimates in synchrony is required. Two alternatives for such "distributed" Kalman filtering have been proposed by Hashemipour *et al* [HRL88] and Rao *et al* [RDWS93]. These are shown schematically in figure 4.3. On the left of the figure is the architecture due to Hashemipour *et al* [HRL88] in which a single "parent" node communicates via two-way links with a set of "child" nodes. The nodes represent either sensors which return raw observation data, or sensors with some computational ability, used to process the observation data "local" to the node's sensor. The parent performs the bulk of the filter computations, with the child nodes either returning sensor data alone, or perhaps performing some local processing before returning a measurement. In the architecture of Rao *et al* [RDWS93] (figure 4.3, right) each node consists

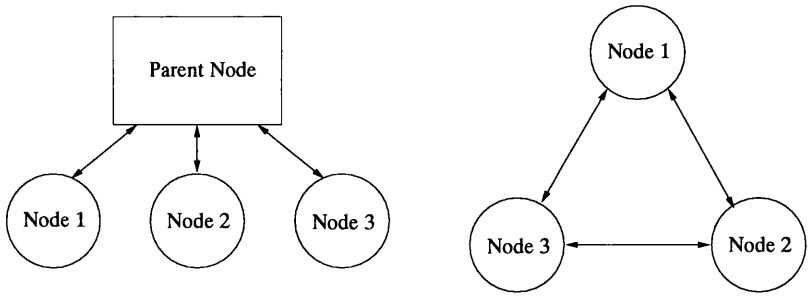


Figure 4.3: Network topologies. Left: That described by Hashemipour *et al* [HRL88]. Right: The topology due to Rao *et al* [RDWS93]. The double headed arrows represent two-way communication connections.

of a sensor with local computation. Each node computes the filter's state evolution model and uses its sensor data to correct the local state predictions in the Kalman filter cycle. Periodically, each node transmits information relating to state (and estimate covariance) updates around the

network to further correct and synchronise estimates between nodes. This topology is known as a "fully decentralised" architecture, as there is no single node upon which the network relies to function. An advantage of the decentralised filter is that the network is robust to node failure; if a single node malfunctions, the other nodes will be unaffected and will continue to estimate as before. However, the overall computational load is greater for this architecture as each individual node must compute predictions and make individual state updates.

Node failure is not the main issue in our application, where computational complexity and system simplicity are more important, so an architecture similar to that of Hashemipour has been chosen for the on-line system. The dead-reckoning filter takes the role of the parent node, receiving sensor data from the odometers and accelerometers in addition to processed data from the vision system which, like the other sensors, is a child node. Each time the vision system acquires an image, it requests a state estimate from the dead-reckoning node in order to predict the position of crop features in the image, and, upon completion of an estimate update using the observed image data, returns information to the dead-reckoning node in the form of a new estimate of tx, Y and Ψ. The overall computational burden is lower than the decentralised system, and also both dead-reckoning and vision are vital to successful operation of the vehicle, so the robustness to node failure offered by the decentralised architecture offers no practical advantage. If the vision system were to fail, crop treatment would become impossible, and without the dead-reckoning system, there would be no estimation of velocity or acceleration, which are required to provide predictions for the vision system.

The architecture of the vehicle's estimation system is outlined in figure 4.4, which shows the vision system and the dead-reckoning filters with their sensor "inputs" in brackets, together with the flow of information between them. The dead-reckoning filter runs at 50 Hz, incorporating information from wheel odometers and inertial sensors (accelerometers) into a state estimate prediction $\hat{\mathbf{x}}_{dr}^{-}(k+1)$ (position, velocity, acceleration) with covariance matrix $\mathbf{P}_{dr}^{-}(k+1)$. Each time the vision system grabs an image from the vehicle's camera, it requests a state prediction from the dead-reckoning filter. The prediction is transformed from the dead-reckoning co-ordinate frame into the vision system co-ordinates by a function $\mathbf{g}(\mathbf{x}_{dr}(k+1))$. The prediction covariance is also projected into the vision system's co-ordinate frame using the matrix of partial derivatives of $\mathbf{g}(\cdot)$. The transformed state prediction is used to generate the expected position of the crop grid in the image, and the transformed covariance is used in a matching and validation strategy which associates features extracted by image processing with the plant positions predicted by the state vector. The matching and validation mechanisms are detailed later in chapter 6. The vision system then takes these matched features and their predictions to pro-

duce a single "pseudo-observation" (denoted $\tilde{\mathbf{z}}(k+1)$ in the diagram, and seen in section 4.6.1 below) and its covariance matrix ($\tilde{\mathbf{R}}(k+1)$) which is then passed back to the dead-reckoning filter to provide a vision-based update of the dead-reckoning estimate.

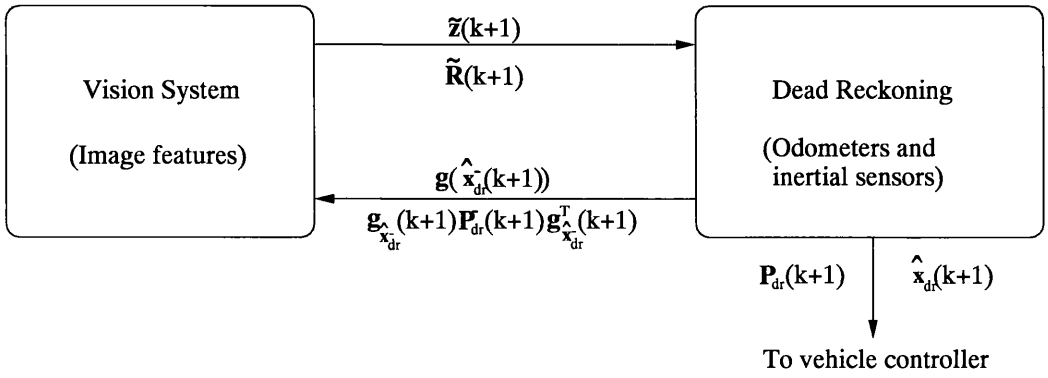


Figure 4.4: Estimation system schematic.

To summarise, in both the on-line and off-line systems, the vision system provides estimates of t_x, Y and Ψ. In the off-line system, predictions for t_x, Y and Ψ are obtained from the state evolution model described in equation 4.1, whilst in the on-line system, they are provided by the dead-reckoning filter. Additionally, in the on-line system, the vision system filter's estimate is passed back to the dead-reckoning estimator which incorporates the new data into its own state estimate. The mechanism which allows this separation of the vision and dead-reckoning filters is the data compression filter [WCD76, SHMB99], the third (and final) alternative formulation of the Kalman filter of interest in this thesis.

4.6.1 The data compression filter

The data compression filter forms a least-squares combination of a set of measurements to produce a composite "pseudo-observation" which is then used in the filter update equations. This pseudo-observation is denoted $\tilde{\mathbf{z}}(k)$, and is simply defined by

$$\tilde{\mathbf{z}}(k) = \mathbf{x}(k), \tag{4.66}$$

where $\mathbf{x}(k)$ is the vision system state vector $[t_x, Y, \Psi]^T$. The $\tilde{\mathbf{z}}(k)$ notation is used to emphasise the way in which the vision data is treated as a sensor, and also to prevent confusion between the vision system state vector $\mathbf{x}(k)$ of the off-line system and the dead-reckoning system state vector $\mathbf{x}_{dr}(k)$, which contains position, velocity and acceleration information.

The following non-linear relationship describes the transformation between the dead-reckoning state vector and pseudo-observation:

$$\tilde{\mathbf{z}}(k) = \mathbf{g}(\mathbf{x}_{dr}(k)), \tag{4.67}$$

whilst the relationship between the pseudo-observation and image features is

$$\mathbf{z}(k,m,n) = \mathbf{h}(\tilde{\mathbf{z}}(k), \bar{r}, \bar{l}, m, n). \tag{4.68}$$

Note that equation 4.68 is equivalent to equation 4.13, with $\tilde{\mathbf{z}}(k)$ substituted for the vision system state co-ordinates t_x, Y and Ψ. In the on-line implementation $\bar{r}$ and $\bar{l}$ are fixed.

From equations 4.67 and 4.68, it follows that the observation function $\mathbf{h}_{dr}(\mathbf{x}_{dr}(k))$ may be written that relates the dead-reckoning state vector $\mathbf{x}_{dr}(k)$ to the image features $\mathbf{z}(k, \bar{r}, \bar{l}, m, n)$:

$$\mathbf{z}(k,m,n) = \mathbf{h}_{dr}(\mathbf{x}_{dr}(k)) = \mathbf{h}(\mathbf{g}(\mathbf{x}_{dr}(k)), \bar{r}, \bar{l}, m, n). \tag{4.69}$$

For the Kalman filter equations, the matrix of partial derivatives of $\mathbf{h}_{dr}(k)$ with respect to $\mathbf{x}_{dr}(k)$ is required. This matrix, $\mathbf{h}_{dr_{\mathbf{x}_{dr}}}(k)$, may be obtained by using the chain rule when differentiating equation 4.69. Using the identities

$$\mathbf{h}_{\tilde{\mathbf{z}}(k)}(k) = \left.\frac{\partial \mathbf{h}(\tilde{\mathbf{z}}(k)), \bar{r}, \bar{l}, m, n}{\partial \tilde{\mathbf{z}}(k)}\right|_{\tilde{\mathbf{z}}(k)=\mathbf{g}(\hat{\mathbf{x}}_{dr}^-)}, \tag{4.70}$$

and

$$\mathbf{g}_{\hat{\mathbf{x}}_{dr}^-(k)}(k) = \left.\frac{\partial \mathbf{g}(\mathbf{x}_{dr}(k))}{\partial \mathbf{x}_{dr}(k)}\right|_{\mathbf{x}_{dr}(k)=\hat{\mathbf{x}}_{dr}^-(k)}, \tag{4.71}$$

$\mathbf{h}_{dr_{\mathbf{x}_{dr}(k)}}(k)$ may be written

$$\mathbf{h}_{dr_{\mathbf{x}_{dr}(k)}}(k) = \mathbf{h}_{\tilde{\mathbf{z}}(k)}(k)\mathbf{g}_{\hat{\mathbf{x}}_{dr}^-(k)}(k). \tag{4.72}$$

The matrix $\mathbf{h}_{\tilde{\mathbf{z}}(k)}$ is that given by equation 4.16 (recall that the pseudo-observation $\tilde{\mathbf{z}}(k)$ is identical to the vision system state vector $\mathbf{x}(k)$).

The following equations (due to Hague [SHMB99]) substitute the expression of equation 4.72 into the information filter (equations 4.40 and 4.41) for the dead-reckoning state estimate $\hat{\mathbf{x}}_{dr}(k+1)$ and covariance $\mathbf{P}_{dr}(k+1)$:

$$\mathbf{P}_{dr}^{-1}(k+1) = (\mathbf{P}_{dr}^-)^{-1}(k+1) + \mathbf{g}_{\hat{\mathbf{x}}_{dr}(k+1)}^T \tilde{\mathbf{R}}^{-1}(k+1)\mathbf{g}_{\hat{\mathbf{x}}_{dr}(k+1)} \tag{4.73}$$

$$\begin{aligned}\mathbf{P}_{dr}^{-1}(k+1)\hat{\mathbf{x}}_{dr}(k+1) = \\ [(\mathbf{P}_{dr}^-)^{-1}(k+1) + \mathbf{g}_{\hat{\mathbf{x}}_{dr}^-(k+1)}^T(k+1)\tilde{\mathbf{R}}^{-1}(k+1)\mathbf{g}_{\hat{\mathbf{x}}_{dr}^-(k+1)}]\hat{\mathbf{x}}_{dr}^-(k+1) + \\ \mathbf{g}_{\hat{\mathbf{x}}_{dr}^-(k+1)}^T(k+1)\tilde{\mathbf{R}}^{-1}(k+1)[\tilde{\mathbf{z}}(k+1) - \mathbf{h}(\mathbf{g}((\hat{\mathbf{x}}_{dr}^-(k+1))))]\end{aligned} \tag{4.74}$$

In the two equations above, the quantity $\tilde{\mathbf{R}}(k)$ is the covariance of the pseudo observation $\tilde{\mathbf{z}}(k)$:

$$\tilde{\mathbf{R}}^{-1}(k) = \mathbf{h}_{\tilde{\mathbf{z}}(k)}(k)^T \mathbf{R}^{-1}(k)\mathbf{h}_{\tilde{\mathbf{z}}(k)}(k), \tag{4.75}$$

and our update equation is

$$\tilde{\mathbf{R}}^{-1}(k)\tilde{\mathbf{z}}(k) = \tilde{\mathbf{R}}^{-1}(k)\mathbf{g}(\hat{\mathbf{x}}_{dr}^{-}(k)) + \mathbf{h}_{\tilde{\mathbf{z}}(k)}(k)^{T}\mathbf{R}^{-1}(k)[\mathbf{z}(k,m,n) - \mathbf{f}(\mathbf{g}(\hat{\mathbf{x}}_{dr}^{-}(k)))]. \quad (4.76)$$

Equations 4.75 and 4.76 effectively separate computations involving the image features $\mathbf{z}(k,m,n)$ from the update of the dead-reckoning state estimate $\hat{\mathbf{x}}_{dr}(k)$. Thus, if a set of S observed features are extracted from the image, where each feature s is matched to a different grid position (m,n), they may be combined into a single pseudo-observation $\tilde{\mathbf{z}}(k)$ as follows where the argument s attached to various quantities denotes that the quantity relates to feature s, as in section 4.3.2:

$$\tilde{\mathbf{R}}^{-1}(k) = \sum_{s=1}^{s=S} \mathbf{h}_{\tilde{\mathbf{z}}(k)}(k,s)\mathbf{R}^{-1}(k,s)\mathbf{h}_{\tilde{\mathbf{z}}(k)}(k,s) \quad (4.77)$$

$$\tilde{\mathbf{R}}^{-1}(k)\tilde{\mathbf{z}}(k) = \tilde{\mathbf{R}}^{-1}(k)\mathbf{g}(\hat{\mathbf{x}}_{dr}^{-}(k)) + \sum_{s=1}^{s=S} \mathbf{h}_{\tilde{\mathbf{z}}(k)}(k,s)^{T}\mathbf{R}^{-1}(k,s)[\mathbf{z}(k,s) - \mathbf{h}(\mathbf{g}(\hat{\mathbf{x}}_{dr}^{-}(k),s))]. \quad (4.78)$$

Equations 4.77 and 4.78 compress all of the image data into a single pseudo-observation with an accompanying covariance matrix, hence the name "data compression filter". Furthermore, this function is performed by the vision sub-system on the left of the schematic diagram of figure 4.4. These quantities are then passed back to the dead-reckoning system which updates the dead-reckoning estimate using equations 4.73 and 4.74.

4.6.2 Observability of the on-line system

Despite the differences between the on-line and off-line systems, the observation function at the heart of both of the vision filters system is the same, i.e. $\mathbf{h}(\mathbf{x}(k))$ =$\mathbf{h}(\tilde{\mathbf{z}}(k))$. To produce a valid pseudo-observation $\tilde{\mathbf{z}}(k)$, the same conditions on the number of matched image features must be met as for the off-line filter. With fixed grid parameters, as in the on-line case, the number of features required is two (see section 4.5).

4.6.3 Estimating $\bar{r}$ and $\bar{l}$ on-line

The estimation of $\bar{r}$ and $\bar{l}$ has yet to be implemented in the on-line system, but there is no theoretical bar to doing so, and the changes required are merely adjustments to the system described above. First, we add the prediction and update of $\bar{r}$ and $\bar{l}$ into the dead-reckoning system, so the state vector $\mathbf{x}_{dr}(k)$ is augmented to include $\bar{r}$ and $\bar{l}$. We then include $\bar{r}$ and $\bar{l}$ in the pseudo-observation vector $\tilde{\mathbf{z}}(k)$ thus

$$\tilde{\mathbf{z}}(k) = \mathbf{x}(k) = [t_x, Y, \Psi, \bar{r}, \bar{l}]^{T}, \quad (4.79)$$

echoing equation 4.66. Finally, the function $\mathbf{g}(x_{dr}(k))$ given in equation 4.67 must be adapted to reflect the changes in $\mathbf{x}_{dr}(k)$ and $\tilde{\mathbf{z}}(k)$. These adapted quantities are then used in the data compression filter as usual.

4.7 Summary

Models for both the off-line and on-line filters have been introduced. In the off-line case, two filters have been examined; one where the crop grid parameters are fixed, and a second where they are estimated. It has been shown that for the second filter, the grid parameters are partially incorruptible, and that noise should only affect these states when either new plants enter the image, or when previously seen plants leave. Both filters satisfy the linearised observability condition devised in chapter 3, provided that a sufficient number of features that match the predicted crop grid positions are extracted from each image. In the case of the fixed parameter filter, two matched features are required from an image, and when the grid parameters are estimated, this figure rises to three. This knowledge may be used to improve the robustness of the system; if the image processing fails to produce a sufficient number of matched features, then observability is not guaranteed and no state update should be attempted with features from that image. The process of feature matching will be discussed in chapter 6.

In the case of the on-line filter, the system architecture has been described, together with the data compression filter which allows separation of vision and dead-reckoning computation. The dead-reckoning system runs at 50 Hz, and passes a prediction to the crop grid tracker each time an image is processed. This prediction is used in the least-squares compression of all the observations extracted by image processing to produce a pseudo-observation with an associated covariance matrix which are passed back to the dead-reckoning filter to further correct the vehicle's position estimate.

The next chapter will show the crop grid tracking filter in action in two cases. The first is an off-line trial on two short image sequences captured in the field, where the results are compared with alternative automatic methods and human assessment. The second case demonstrates the filter running live on the vehicle in a simplified indoor test-bed environment. Chapter 8 contains experimental data from the algorithm running outdoors in the field.

Chapter 5

Off-line and test-bed navigation trials

To investigate the viability of the crop grid tracker prior to testing in the field, two sets of experiments were performed. The first experiments ran off-line on a desktop workstation and, in the absence of veridical trajectory information, compared the position estimates from the grid tracker algorithm with human assessment of the crop grid position and output from the previous algorithm used on the vehicle for visual feedback (based on the Hough transform, appendix A)[MB95]. For these experiments, two short image sequences captured from the vehicle were used. The results indicate that the crop grid position estimates generated by the tracking algorithms outlined in the previous two chapters compare more favourably with human assessment of crop grid position than the Hough transform output [SMHB98]. This suggests that the extended Kalman filter algorithms will perform well on-line, with the further advantage over the Hough transform algorithm [MB95] (appendix A) that forward distance is estimated and a measure of certainty in the position estimate is provided which allows natural integration with the vehicle's dead-reckoning system (as described previously in section 4.6). In addition, as will be seen in chapter 7, the use of the crop grid model facilitates segmentation of the image into regions of crop, weed and soil. The experimental results also show some quantitative advantage in estimating the crop grid parameters $\bar{r}$ and $\bar{l}$.

The second set of experiments were run with the autonomous vehicle system in a simplified test-bed environment. This environment, a large black mat with a grid of white circles painted on it to represent the crop, provides a much simplified image processing problem for the vision system, and it also allows straightforward measurement of system performance. Experiments on this test-bed provide a useful indication of whether the on-line system is likely to function with real crop in the field because the basic grid structure used for navigation is emulated. The test-bed offers a number of advantages over the field environment for testing prototype systems. Firstly, the image processing is less error prone, because of the strong contrast between plant matter (the white circles) and soil (the black mat), and also, because the circles are 2D objects ly-

ing on the ground plane, the projection errors discussed in chapter 2 do not occur[1]. Performance measurements are straightforward because the grid of circles has been painted in a known position, and the grid pattern is more regular than that typically found in real crop. The experiment consists of the vehicle leaving a marker trail along the mat, and then measuring the position of this trail being relative to the central row of circles. The measured trajectory is then compared with the estimated trajectory from the vehicle's navigation system. Because the position of the central row is precisely known (unlike real crop, where the rows are not straight), such measurement may be made with reasonable accuracy indoors. A final advantage of course, is that the test-bed may be used at any time without the expense of growing and maintaining a field of real plants.

The measurements resulting from the test-bed experiments show that the vehicle can navigate with sufficient accuracy to allow crop-row following with little risk of damaging the crop. Experiments measuring performance on-line in the field are described in chapter 8.

5.1 Off-line experiments

Two short sequences of 20 near infra-red images were digitised from video stock collected from the experimental vehicle during the Summer of 1997. In both sequences, the vehicle was instructed to follow the crop rows at a constant velocity of $1ms^{-1}$. The image sequences were analysed using the following methods:

1. The crop grid tracker with fixed grid parameters (AUTO).

2. The crop grid tracker using the human selected crop plant centres as input features (fixed grid parameters) (SEMI).

3. The crop grid tracker with grid parameter estimation (AUTO2)

4. The crop grid tracker using the human selected input features (estimated grid parameters) (SEMI2).

5. Human assessment of the model position; a mouse-driven program was designed to allow the user to place the crop pattern on each image in the sequence. Data from three different people was collected (HUMAN 1–3).

6. The Hough transform algorithm developed by Marchant and Brivot [MB95], which produces estimates of t_x and Ψ alone (HOUGH).

[1]The "virtual ground plane" discussed in section 2.3.8, figure 2.19 is not required

In order to emphasise the effects of estimating the grid parameters, The AUTO and AUTO2 algorithms are analysed separately. In both AUTO and AUTO2, image features are selected automatically using data association techniques, and these can be compared with SEMI and SEMI2, where the crop plant features were selected by hand. Data association will be covered in the next chapter, together with an algorithm for initialising the trackers.

5.1.1 Tracking with fixed grid parameters (AUTO)

The Kalman filter with fixed grid parameters was tested first, with row spacing $\bar{r} = 475mm$ and line spacing $\bar{l} = 450mm$. These two figures were calculated as the mean spacings in the bed of plants filmed in the test footage; they were given as parameters for the AUTO and SEMI algorithms and also used to set up the grid template for generation of the HUMAN data sets. The $\bar{r}$ parameter value was used to construct the row template for the HOUGH algorithm (appendix A). Figures 5.1 and 5.2 show the state trajectories from the first and second image sequences respectively. The left-hand column plots the three human responses, whilst the right hand column shows the equivalent automatic results. Note that there is no Y estimate available from the HOUGH algorithm, and that the resolution of the Hough transform is limited to $16mm$ for t_x and 0.5^o for Ψ.

When ground truth trajectories are unavailable, as in this case, quantitative analysis of the accuracy of a tracking system is difficult to perform because it is not known how the errors are distributed between the automatic methods and the human assessment that is used as a substitute for strict veridical measurements. To provide some measure of performance, an approach has been taken which assumes errors are equally distributed between the automatic and human approach. By taking each set of results for t_x,Y and Ψ from the experiments conducted and pairing the corresponding data sets, scatter plots (like figure 5.3) can be constructed. If a pair of algorithms agree exactly, then the points in the scatter diagram will lie on the line $x = y$. Taking this as the ideal response, a measure of how far a pair of algorithms depart from the ideal may be found by taking the root mean-square differences between feature points in the scatter plot and the nearest (in Euclidean terms) point on the line $x = y$. These values are tabulated for each of the state variables and algorithm pairs in tables 5.1 – 5.6. It should be stressed that the $x = y$ "ideal" does not mean that a pairing is correct, but that the two sets are consistent, so the larger this measure, the greater the inconsistency between them.

The use of consistency measures to assess algorithm performance in the absence of ground truth data is not unprecedented in the machine vision literature. Torr and Zisserman [TZ97] degrade the quality of image pairs (using lossy JPEG compression) and characterise the effects of the degradation on the estimation of the fundamental matrix for the image pairs. In the exper-

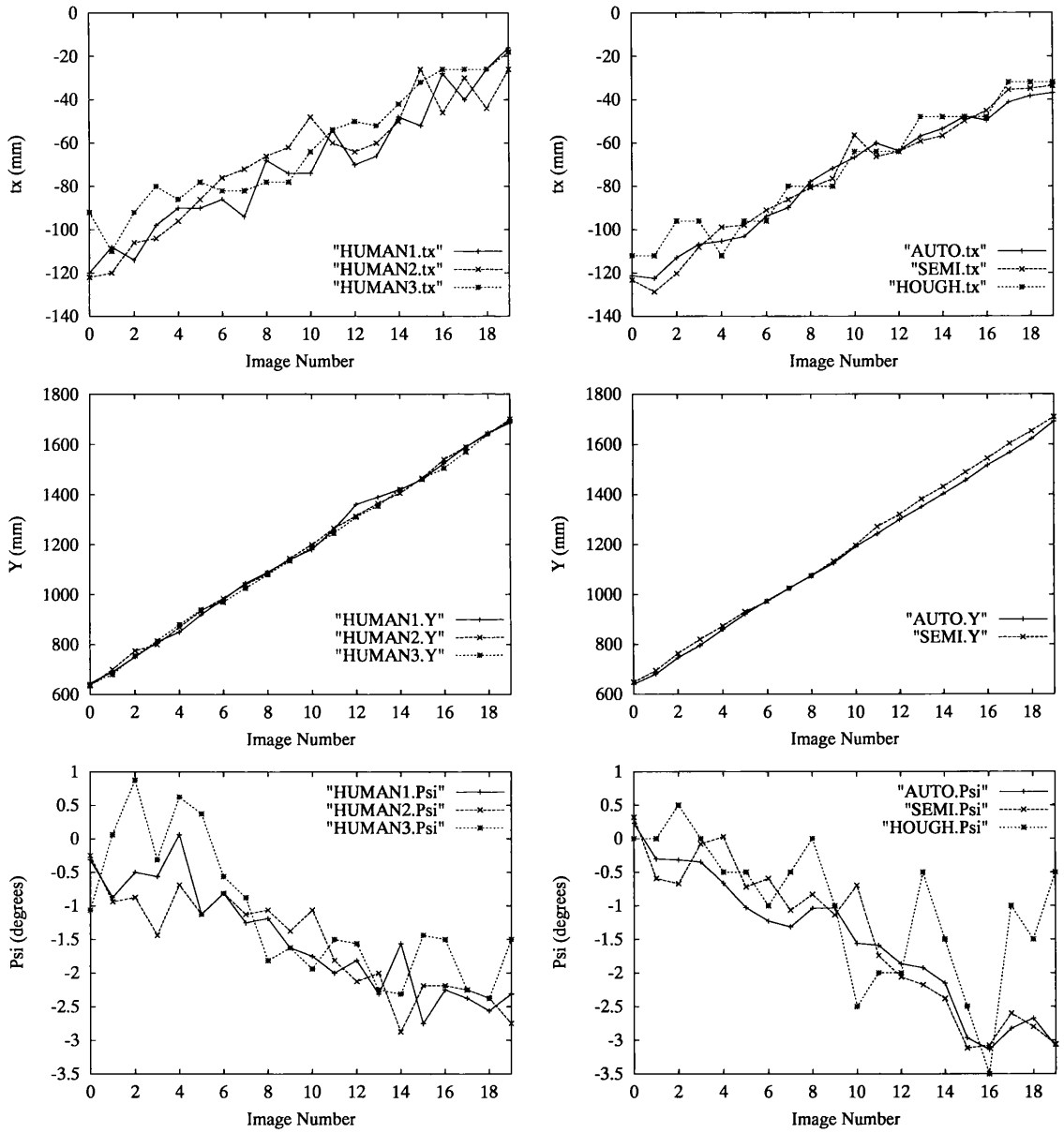


Figure 5.1: Trajectories of the state variables for sequence 1. In the left-hand column are human assessments, and in the right the algorithm output (note that HOUGH does not produce a Y output). The negative values of t_x indicate that the vehicle was to the left of the planting pattern, but moving toward it.

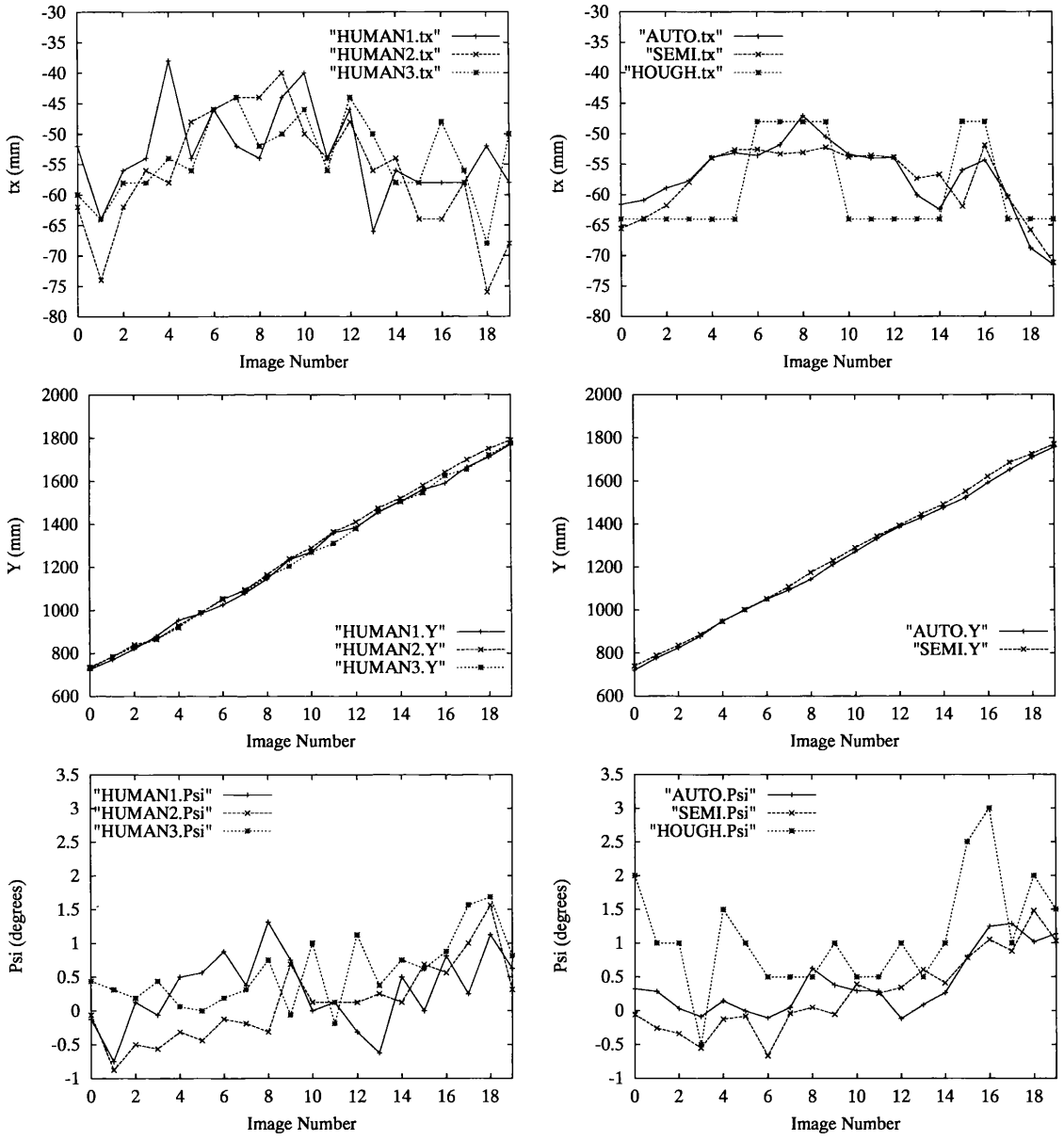


Figure 5.2: Trajectories of the state variables for sequence 2. In the left-hand column are human assessments, and in the right the algorithm output (note that HOUGH does not produce a Y output).

iments they conduct, the ground truth 3D structure of the imaged scene is known in only one of their four examples. In the other three cases, they use the fundamental matrix and supporting matches extracted from the original image pair as ground truth for comparison with the estimates and matches extracted from the degraded pairs.

Leclerc *et al* [LLF99] propose a self-consistency scheme for evaluating a point correspondence algorithm in the absence of ground truth. By taking a number of image pairs of the same scene, a set of image feature matches may be generated for a particular 3D scene feature (one match per image pair that the scene feature is visible in). Each matched pair is then used to reconstruct the scene point (or back-project it into an image plane), and the consistency of the reconstructed point position produced by each match pair is studied. The authors propose distributions that may be calculated from the match consistencies and relate these to the absolute accuracy of the algorithm.

Although the self-consistency approaches of the above pieces of work [TZ97, LLF99] provide insight into the performance of the algorithms under analysis, we must remember that consistency is not the same as correctness. An algorithm may perform in a highly consistent manner across many data sets, but if its output does not reflect the ground truth, then the algorithm is not suitable for the task at hand. In our experiments, we compare the output of the crop grid tracking algorithms with the HOUGH algorithm which is known to perform in the field in a satisfactory (if limited) manner, and with the assessment of three independent human observers.

By generating self-consistency measures, Leclerc *et al* [LLF99] and Torr and Zisserman [TZ97] have benefited from being able to compare like with like. Unfortunately, the various methods used for assessment of the crop grid position produce different forms of output. The HUMAN data sets contain estimates of t_x, Y and Ψ, whilst the HOUGH algorithm only produces t_x and Ψ. The AUTO and SEMI algorithms produce all three position parameters, and also covariance matrices describing the uncertainty in those parameters. AUTO2 and SEMI2 also deliver estimates of the grid parameters $\bar{r}$ and $\bar{l}$. These differences in dimensionality for the various assessments have lead to the simple parameter by parameter comparisons used here.

From the figures in tables 5.1 – 5.6 and perusal of the trajectory plots, figures 5.1 and 5.2, three main conclusions may be drawn:

1. Human assessments are not wholly consistent. The table elements referring to the similarity between HUMAN data set pairs contain figures that are not zero. Unsurprisingly, different people make differing assessments of the model position.

2. The various algorithm estimates are as consistent with the human results as the human re-

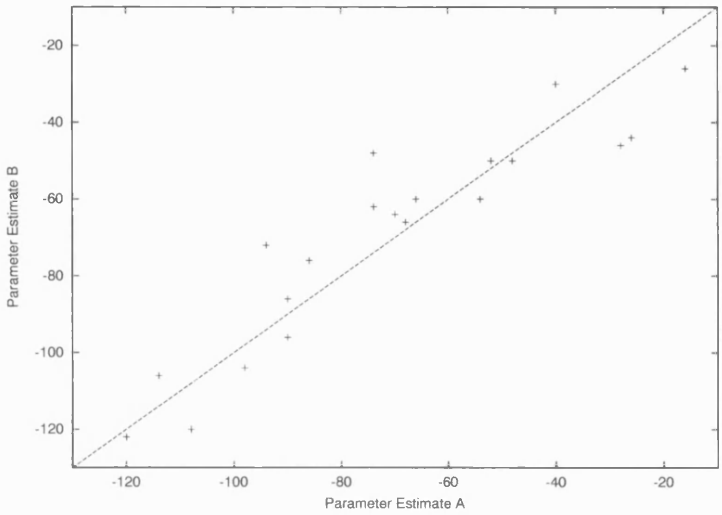


Figure 5.3: A scatter plot: the points show the scatter data (two independent measures of the same parameter, in this particular case t_x), while the solid line plots the line of perfect consistency between the data sets. A measure of consistency is the root mean square distance of the scattered data points from the line.

sults are with each other. In some cases, notably the t_x estimate of HUMAN 2, the human observation is more consistent with the automatic and semi-automatic methods than with the other humans. Importantly, the new method (AUTO) has similar r.m.s. measures of consistency to the HOUGH method, which has operated successfully on the vehicle. For the measurement of Ψ, the AUTO method is more consistent with human assessment than HOUGH. This is reflected in the plots of figures 5.1 and 5.2.

3. The fully automatic algorithm performs comparably with the semi-automatic algorithm. The similarity measures between the two are given in the tables. In the case of t_x and Ψ, the similarity figures are a fraction of the resolution of the HOUGH algorithm, so it may be assumed that errors in the automatic method would not lead to any operational degradation. The mean difference of 14.1 mm on a measurement of Y ranging from 600 – 1700 mm is also very small.

As has been noted, the Kalman filter algorithms not only produce estimates of state value, but also confidence measures reflecting the uncertainty on the current estimates, the covariance matrix $\mathbf{P}(k)$ in the filter equations. For each sequence, the updated state covariance matrix $\mathbf{P}(k)$ at every image was logged. The diagonal terms of $\mathbf{P}(k)$ give the variance on each state estimate (the symmetric off-diagonal terms reflecting covariance between state variables), and hence the standard deviation for the estimates may be obtained by taking the square root of the variances.

	HUMAN 1	HUMAN 2	HUMAN 3	AUTO	SEMI	HOUGH
HUMAN 1	0	–	–	–	–	–
HUMAN 2	9.24	0	–	–	–	–
HUMAN 3	9.41	10.08	0	–	–	–
AUTO	6.81	7.84	11.47	0	–	–
SEMI	7.25	7.51	11.61	3.40	0	–
HOUGH	8.08	8.47	8.95	5.38	6.51	0

Table 5.1: Root-mean square differences on the t_x estimate in mm for sequence 1. '-' indicates that the value can be found in the lower half of table.

	HUMAN 1	HUMAN 2	HUMAN 3	AUTO	SEMI	HOUGH
HUMAN 1	0	–	–	–	–	n/a
HUMAN 2	11.7	0	–	–	–	n/a
HUMAN 3	13.5	10.6	0	–	–	n/a
AUTO	14.8	11.3	7.9	0	–	n/a
SEMI	11.9	9.5	12.5	15.2	0	n/a
HOUGH	n/a	n/a	n/a	n/a	n/a	n/a

Table 5.2: Root-mean square differences on the Y estimate in mm for sequence 1. 'n/a' indicates Y is not estimated by algorithm HOUGH.

	HUMAN 1	HUMAN 2	HUMAN 3	AUTO	SEMI	HOUGH
HUMAN 1	0	–	–	–	–	–
HUMAN 2	0.339	0	–	–	–	–
HUMAN 3	0.508	0.592	0	–	–	–
AUTO	0.305	0.373	0.613	0	–	–
SEMI	0.332	0.382	0.629	0.245	0	–
HOUGH	0.603	0.677	0.714	0.677	0.735	0

Table 5.3: Root-mean square differences on the Ψ estimate in degrees for sequence 1. The figures indicate that the set least consistent with the other data is from algorithm HOUGH, and this is reflected in the plots of figure 5.1.

	HUMAN 1	HUMAN 2	HUMAN 3	AUTO	SEMI	HOUGH
HUMAN 1	0	–	–	–	–	–
HUMAN 2	6.73	0	–	–	–	–
HUMAN 3	5.42	5.34	0	–	–	–
AUTO	5.73	4.55	4.91	0	–	–
SEMI	5.83	4.59	4.77	2.08	0	–
HOUGH	8.12	7.05	6.50	4.72	5.14	0

Table 5.4: Root-mean square differences on the t_x estimate in mm for sequence 2. '-' indicates that the value can be found in the lower half of table.

	HUMAN 1	HUMAN 2	HUMAN 3	AUTO	SEMI	HOUGH
HUMAN 1	0	–	–	–	–	n/a
HUMAN 2	16.3	0	–	–	–	n/a
HUMAN 3	14.2	16.6	0	–	–	n/a
AUTO	11.8	21.6	12.1	0	–	n/a
SEMI	12.9	12.1	12.4	13.0	0	n/a
HOUGH	n/a	n/a	n/a	n/a	n/a	n/a

Table 5.5: Root-mean square differences on the Y estimate in mm for sequence 2. 'n/a' indicates Y is not estimated by algorithm HOUGH.

	HUMAN 1	HUMAN 2	HUMAN 3	AUTO	SEMI	HOUGH
HUMAN 1	0	–	–	–	–	–
HUMAN 2	0.469	0	–	–	–	–
HUMAN 3	0.509	0.460	0	–	–	–
AUTO	0.400	0.362	0.318	0	–	–
SEMI	0.494	0.260	0.359	0.261	0	–
HOUGH	0.827	0.862	0.650	0.655	0.749	0

Table 5.6: Root-mean square differences on the Ψ estimate in degrees for sequence 2. The figures indicate that the set least consistent with the other data is from algorithm HOUGH, and this is reflected in the plots of figure 5.2.

Tables 5.7 and 5.8 show the average standard deviation of the state variables for each sequence. It can be seen from comparison of tables 5.7 and 5.8 with tables 5.1 – 5.6 that, typically, the standard deviations produced by the filters are lower than the r.m.s. differences between the filter estimates (either AUTO or SEMI) and the HUMAN data sets, which serve as the ground truth estimates. This would indicate that the extended Kalman filter algorithms are over-optimistic in their assessment of estimate accuracy. This property of extended Kalman filters is well known [May90] and can sometimes lead to loss of track. However this has not occurred in the two test sequences, nor, as will be seen below, in the on-line trials (section 5.2.1).

	AUTO	SEMI
σ_{t_x} (mm)	5.65	5.52
σ_Y (mm)	10.3	10.42
σ_Ψ (degrees)	0.443	0.433

Table 5.7: Filter estimate standard deviations from the AUTO and SEMI algorithms, sequence 1 data.

	AUTO	SEMI
σ_{t_x} (mm)	5.39	5.29
σ_Y (mm)	9.85	10.1
σ_Ψ (degrees)	0.413	0.451

Table 5.8: Filter estimate standard deviations from the AUTO and SEMI algorithms, sequence 2 data.

5.1.2 Tracking whilst estimating grid parameters (AUTO2)

The second Kalman filter algorithm not only tracks the grid position, but also estimates the crop grid spacing parameters $\bar{r}$ and $\bar{l}$ as it traverses the field, allowing local variations in plant spacing to be accounted for. The algorithm was run fully automatically (AUTO2) and with hand-picked image features (SEMI2) on the two short test image sequences, and the results compared with the HUMAN1 – HUMAN3 and HOUGH estimates as above. The root mean square differences on the t_x and Ψ estimates were found to be similar to those for the Kalman filter algorithm with fixed grid parameters; however the Y estimation appeared to be considerably worse, especially for the first of the two sequences. This is illustrated in figure 5.4, where the AUTO2 and SEMI2 tracks for sequence 1 are plotted on the same axes as the three human assessments.

As can be seen in the diagram, the automatically and semi-automatically generated esti-

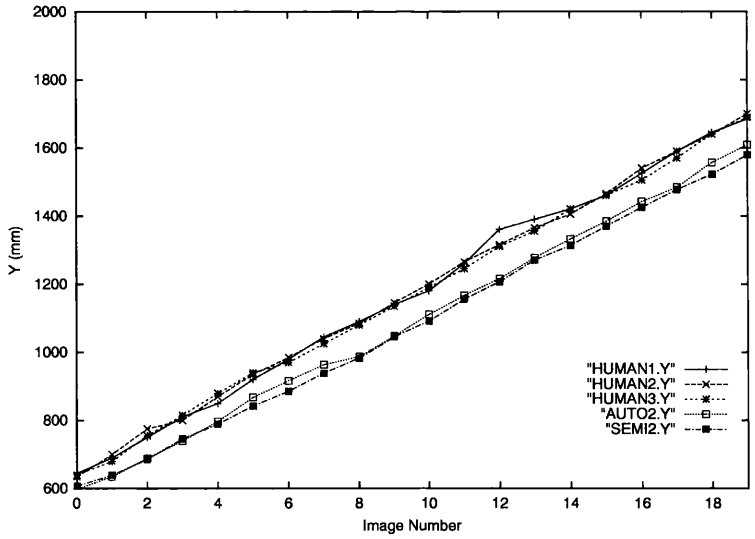


Figure 5.4: The Y estimates from AUTO2 and SEMI2, together with the human assessments, for sequence 1.

mates of Y position (the two lower curves) are somewhat different from the human assessments (the three upper curves), and are also diverging slightly. A similar plot for the original algorithms with fixed $\bar{r}$ and $\bar{l}$, AUTO and SEMI, shows all the curves in good agreement (figure 5.5). It would thus appear at first sight that estimating the grid parameters as the vehicle traverses the crop has resulted in a system with poorer performance. In fact, further investigation reveals that this is not the case, but that the human "ground truth" assessments have been made on an unsuitable basis.

The forward distance estimate is the sum of the distance of the bottom plant in the current image from the vehicle (Y as marked in figure 4.1) and an increment of $\bar{l}(k)$ each time a plant is passed, as seen in equation 4.6. Evidently, if $\bar{l}(k)$ is fixed, the increment to the total forward distance will be constant. If the value of $\bar{l}(k)$ estimated by the filter is lower (or higher) than the value given to the fixed parameter algorithm, then the forward distance estimates will differ systematically. Also, if two grids with different spacing parameters are fitted to the same image data, the position and orientation of best fit may be different for each grid (this is illustrated later in figure 5.8).

From figures 5.4 and 5.5 it is clear that the estimated values of $\bar{l}$ must be lower than the $450mm$ (the mean spacing within the crop row for the whole crop bed) given to the fixed grid tracker. Because the camera is calibrated, it is possible to obtain an approximate measure of the mean grid parameters for each of the short sequences (which may differ from the overall mean values for the entire bed as used in the AUTO and SEMI algorithms) by selecting the crop plant positions by hand from the images and calculating, via the camera calibration, the dis-

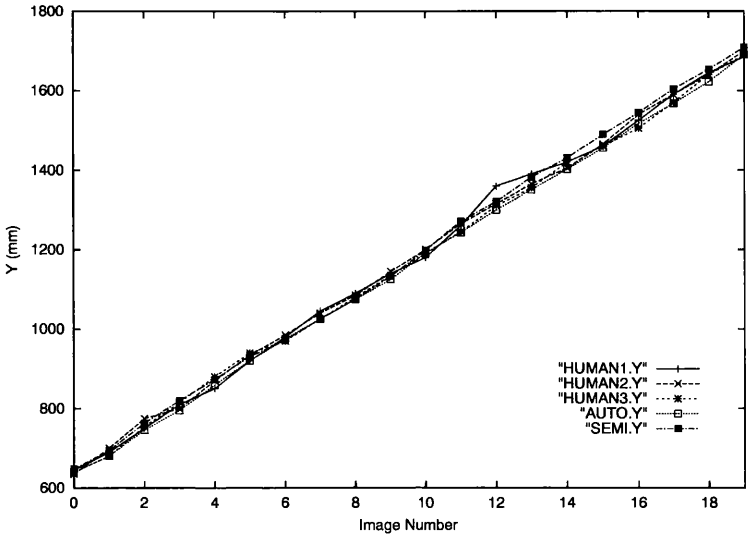


Figure 5.5: The Y estimates from AUTO and SEMI, together with the human assessments, for sequence 1.

tance on the ground plane between plants. Figures 5.6 and 5.7 show the $\bar{r}$ and $\bar{l}$ estimates for the first and second sequences, together with the hand-measured means for each image. As

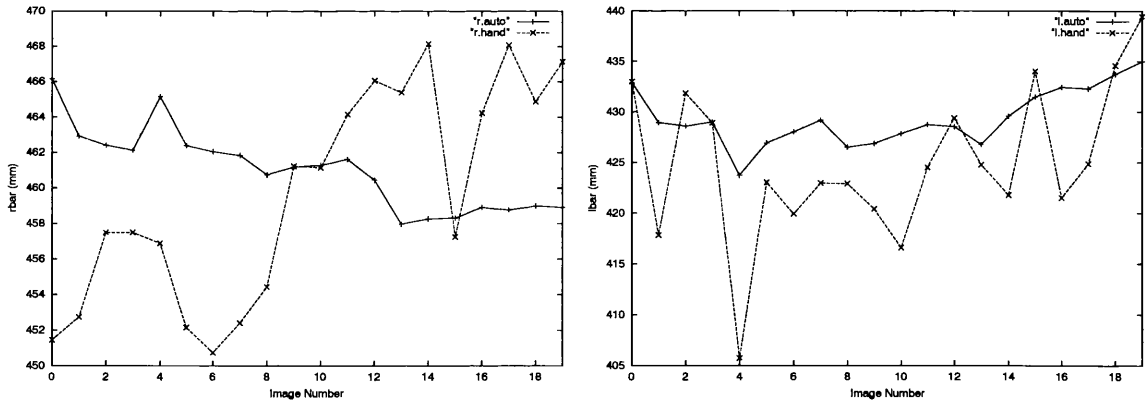


Figure 5.6: Estimated and measured $\bar{r}$ (left) and $\bar{l}$ (right) for the first sequence. The solid line marks the estimated values, and the dashed line those measured by hand.

can be seen in the plots, the estimates and hand-measurements are not entirely in agreement. The consistency figures are given below in tables 5.13, 5.14, 5.18 and 5.19, and they show that the similarity measures between the HUMAN and AUTO2 assessments of the grid parameters reflect these differences, although the figures are small when compared to the parameter values. The hand-measured sequence means also deviate from the crop bed means of $\bar{r} = 475mm$ and $\bar{l} = 450mm$; the mean hand-measured parameters from each sequence are given in table 5.9. Upon comparing the figures in table 5.9 with the fixed parameters in the algorithm AUTO

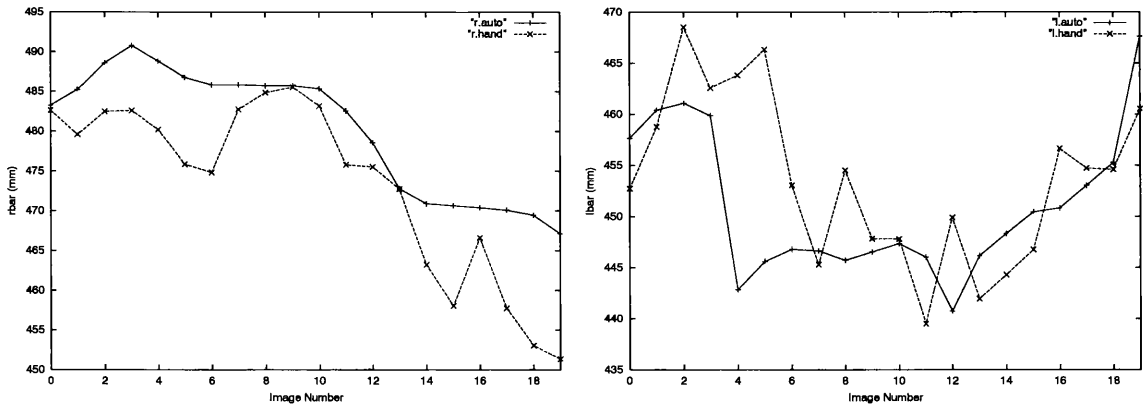


Figure 5.7: Estimated and measured $\bar{r}$ (left) and $\bar{l}$ (right) for the second sequence. The solid line marks the estimated values, and the dashed line those measured by hand.

Sequence	mean $\bar{r}(mm)$	mean $\bar{l}(mm)$
1	460	425
2	473	454

Table 5.9: The hand-measured sequence means of $\bar{r}$ and $\bar{l}$.

($\bar{r} = 475mm$, $\bar{l} = 450mm$), the reason for the difference in Y estimate on sequence 1 (figure 5.4) becomes clear; the mean $\bar{l}$ figure for this sequence is $25mm$ lower than that of the fixed grid used in the generation of the data sets HUMAN1 – HUMAN3. As noted above, a change in $\bar{l}$ will lead to a different position of best fit, hence the initial offset between the two sets of traces in figure 5.4, and also a different rate of growth in the total Y estimate, hence the slight divergence of the traces. By the end of the sequence, the two AUTO and AUTO2 estimates of Y differ by $82mm$, which has accumulated over little more than one metre of travel. The estimated crop grid positions for the first image of sequence 1 are illustrated in figure 5.8, where they are superimposed on the image, and the different positions of best fit can be seen clearly.

To re-assess the performance of the AUTO2 and SEMI2 algorithms, a human assessed data set, named HUMAN, was produced for each run with the grid parameters set to those measured from the sequences (table 5.9). The resulting traces are plotted alongside the AUTO2 and SEMI2 algorithm output in figures 5.9 and 5.10. Tables 5.10 – 5.19 give the consistency figures between the various algorithms, including a comparison with the fixed parameter tracker AUTO (in which $\bar{r}$ and $\bar{l}$ are fixed at the values $\bar{r} = 475mm$, $\bar{l} = 450mm$ as before).

Tables 5.10 – 5.12 show the consistency measures between the HUMAN assessment, the AUTO2 and SEMI2 grid parameter estimators, and the fixed parameter AUTO algorithm, all for

Figure 5.8: Estimates from AUTO and AUTO2. AUTO grid points plotted in white, AUTO2 in black. Owing to the perspective projection, it is easiest to see the difference in the estimated grid positions at the bottom of the image.

	HUMAN	AUTO2	SEMI2	AUTO
HUMAN	0	–	–	–
AUTO2	7.82	0	–	–
SEMI2	9.04	3.26	0	–
AUTO	10.06	4.51	3.44	0

Table 5.10: Root-mean square differences on the t_x estimate in mm for sequence 1. '-' indicates that the value can be found in the lower half of table. The HUMAN data set is manual assessment of crop grid position using a template whose $\bar{r}$ and $\bar{l}$ parameters are equal to the sequence means from table 5.9.

	HUMAN	AUTO2	SEMI2	AUTO
HUMAN	0	–	–	–
AUTO2	9.7	0	–	–
SEMI2	14.5	12.34	0	–
AUTO	46.9	49.06	58.67	0

Table 5.11: Root-mean square differences on the Y estimate in mm for sequence 1. As can be seen, the least consistent data set the the original Kalman filter algorithm AUTO, where the grid parameters are inappropriately set.

	HUMAN	AUTO2	SEMI2	AUTO
HUMAN	0	–	–	–
AUTO2	0.338	0	–	–
SEMI2	0.478	0.277	0	–
AUTO	0.340	0.268	0.276	0

Table 5.12: Root-mean square differences on the Ψ estimate in degrees for sequence 1.

	HUMAN	AUTO2	SEMI2	AUTO
HUMAN	0	–	–	n/a
AUTO2	5.49	0	–	n/a
SEMI2	4.06	3.10	0	n/a
AUTO	n/a	n/a	n/a	n/a

Table 5.13: Consistency measures on estimates of $\bar{r}$ in mm for sequence 1.

	HUMAN	AUTO2	SEMI2	AUTO
HUMAN	0	–	–	n/a
AUTO2	5.10	0	–	n/a
SEMI2	6.13	7.64	0	n/a
AUTO	n/a	n/a	n/a	n/a

Table 5.14: Consistency measures on estimates of $\bar{l}$ in mm for sequence 1.

	HUMAN	AUTO2	SEMI2	AUTO
HUMAN	0	–	–	–
AUTO2	5.56	0	–	–
SEMI2	5.27	2.40	0	–
AUTO	6.20	2.46	2.60	0

Table 5.15: Root-mean square differences on the t_x estimate in mm for sequence 2.

	HUMAN	AUTO2	SEMI2	AUTO
HUMAN	0	–	–	–
AUTO2	17.6	0	–	–
SEMI2	17.5	8.0	0	–
AUTO	14.1	13.7	11.9	0

Table 5.16: Root-mean square differences on the Y estimate in mm for sequence 2. In this case, the AUTO algorithm is marginally more similar with HUMAN assessment that the AUTO2 and SEMI2 algorithms.

	HUMAN	AUTO2	SEMI2	AUTO
HUMAN	0	–	–	–
AUTO2	0.518	0	–	–
SEMI2	0.333	0.362	0	–
AUTO	0.495	0.107	0.371	0

Table 5.17: Root-mean square differences on the Ψ estimate in degrees for sequence 2.

	HUMAN	AUTO2	SEMI2	AUTO
HUMAN	0	–	–	n/a
AUTO2	5.97	0	–	n/a
SEMI2	4.35	3.90	0	n/a
AUTO	n/a	n/a	n/a	n/a

Table 5.18: Consistency measures on estimates of $\bar{r}$ in mm for sequence 2. 'n/a' indicates that the algorithm AUTO does not estimate this parameter.

	HUMAN	AUTO2	SEMI2	AUTO
HUMAN	0	–	–	n/a
AUTO2	5.79	0	–	n/a
SEMI2	8.03	6.30	0	n/a
AUTO	n/a	n/a	n/a	n/a

Table 5.19: Consistency measures on estimates of $\bar{l}$ in mm for sequence 2.

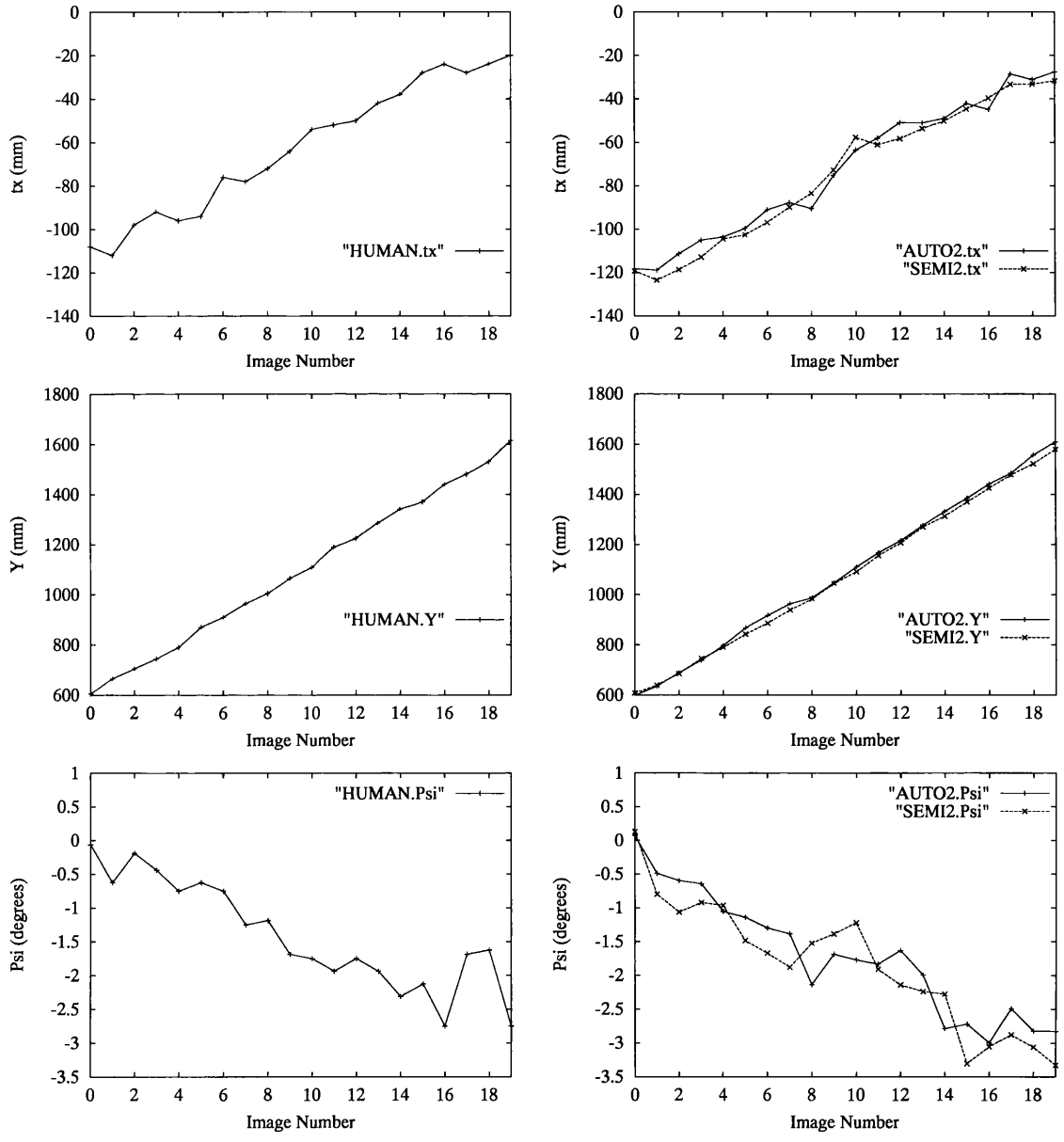


Figure 5.9: State trajectories whilst estimating the grid parameters, sequence 1.

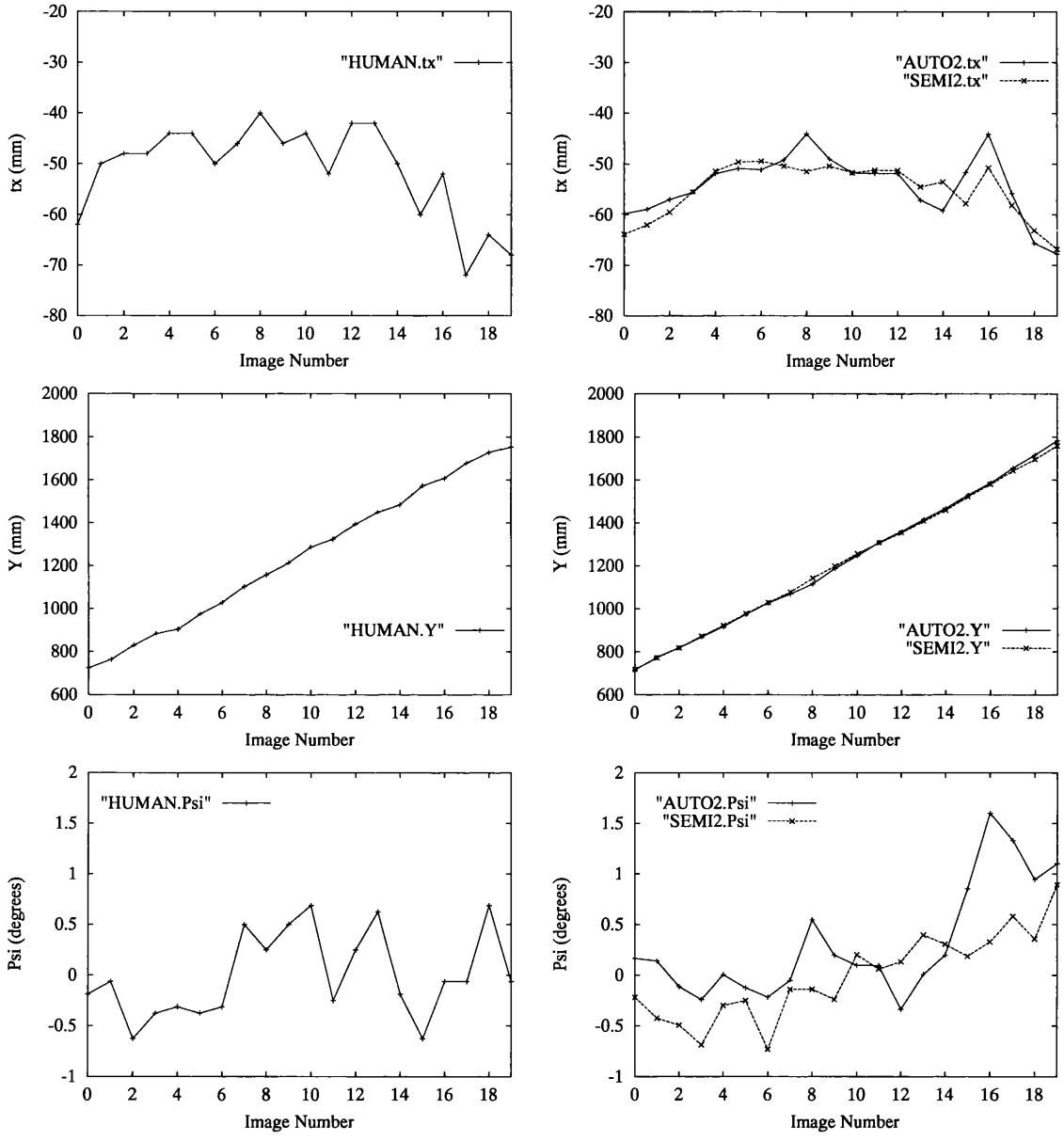


Figure 5.10: State trajectories whilst estimating the grid parameters, sequence 2.

sequence 1. In the case of AUTO, the parameters were set to the crop bed means of $\bar{r} = 475mm$, $\bar{l} = 450mm$ – this is to illustrate any advantage there may be in allowing the parameters to vary. Comparison of AUTO2, SEMI2 and AUTO shows that they perform comparably with each other and HUMAN for both t_x and Ψ (tables 5.10 and 5.12), and in each case AUTO2 shows most agreement with the HUMAN data. The advantage of allowing the grid parameters to vary is clearly illustrated in the estimation of forward distance Y (table 5.11); with an appropriately sized grid, the HUMAN assessment of forward position is considerably more consistent with the AUTO2 and SEMI2 algorithms than the AUTO algorithm. The measure of $14.5mm$ between HUMAN and AUTO2 denotes a much greater consistency than the $46.9mm$ between HUMAN and AUTO. At the end of the sequence, the Y estimates produced by HUMAN and AUTO differ by $76mm$, whilst the difference between the HUMAN and AUTO2 estimates is only $-9mm$, which further reflects the improved tracking performance of AUTO2.

For sequence 2, the true grid parameters are very similar to those given to the AUTO algorithm. The consistency measures are given in tables 5.15 – 5.19 and show a great similarity between all three algorithms and the HUMAN assessment. In fact on the Y estimate the AUTO algorithm is slightly more consistent with HUMAN assessment than AUTO2 and SEMI2, although only marginally so ($3mm$). The same is true for the Ψ estimate, but again the difference is small.

In short, by comparing the AUTO2 algorithm, which estimates local variation in the grid parameters, with human assessments and the original fixed parameter algorithm AUTO, it has been shown that allowing the grid parameters to vary has advantages. When the true grid parameters are very different from those given to AUTO, as is the case for image sequence 1, the AUTO2 algorithm is much better at estimating forward distance than the AUTO algorithm. However, when the actual grid parameters are close to those given to the AUTO algorithm (image sequence 2), the tracking results are very similar for all algorithms. As is the case with the AUTO algorithm, using hand selected features has little bearing on the trajectories, as the similarities between the results for AUTO2 and SEMI2 bear out.

Finally, it is worth noting at this point that both sequence 1 and sequence 2 were captured from different parts of the same bed of crop, yet the variation in grid parameters between the two sequences is great enough to make a significant difference to the tracking performance when the grid parameters are fixed. By estimating the grid parameters, performance is more consistent with the underlying ground truth (as given by human assessment).

As with the AUTO and SEMI algorithms, it is possible to analyse the filter's assessment of its own accuracy in terms of the average standard deviations over each sequence (tables 5.20 and

5.21). Once more, the algorithms are generally more optimistic than comparison with HUMAN assessment might justify, but this has not caused loss of track.

	AUTO2	SEMI2
σ_{t_x} (mm)	5.61	5.67
σ_Y (mm)	13.48	12.95
σ_Ψ (degrees)	0.425	0.448
$\sigma_{\bar{r}}$ (mm)	3.52	3.08
$\sigma_{\bar{l}}$ (mm)	4.07	3.49

Table 5.20: Estimated standard deviations obtained from the filters in the AUTO2 and SEMI2 algorithms for data from sequence sequence 1.

	AUTO2	SEMI2
σ_{t_x} (mm)	5.45	5.31
σ_Y (mm)	13.86	14.57
σ_Ψ (degrees)	0.415	0.455
$\sigma_{\bar{r}}$ (mm)	4.15	4.72
$\sigma_{\bar{l}}$ (mm)	4.89	6.02

Table 5.21: Estimated standard deviations obtained from the filters in the AUTO2 and SEMI2 algorithms for data from sequence sequence 2.

5.2 Test-bed experiments

After showing the viability of the crop grid tracking algorithms on short, off-line data sequences, experiments were conducted to test the performance of the vision system in tandem with the vehicle's dead-reckoning system on-line. Currently only the algorithm AUTO which does not estimate the row spacing parameters has been implemented in the on-line system. In order to avoid difficulties with the image processing and also mechanical problems such as wheel slip, an indoor test-bed has been designed to simplify testing. Figure 5.11 shows the autonomous vehicle on the test-bed, which is simply a large black mat with a set of white circles painted on it in a near regular grid pattern[2]. These circles represent the crop plants. The strong contrast between the white circles and the black mat ensures success of the thresholding process in extracting the "crop plants" from the images, and the fact that the image features do lie accurately

[2]There are some irregularities in the pattern owing to inaccuracies in painting

on the ground plane removes the errors caused by the perspective projection effects that were discussed in section 2.3.8, and the "virtual ground plane" is set to coincide with the real ground plane.

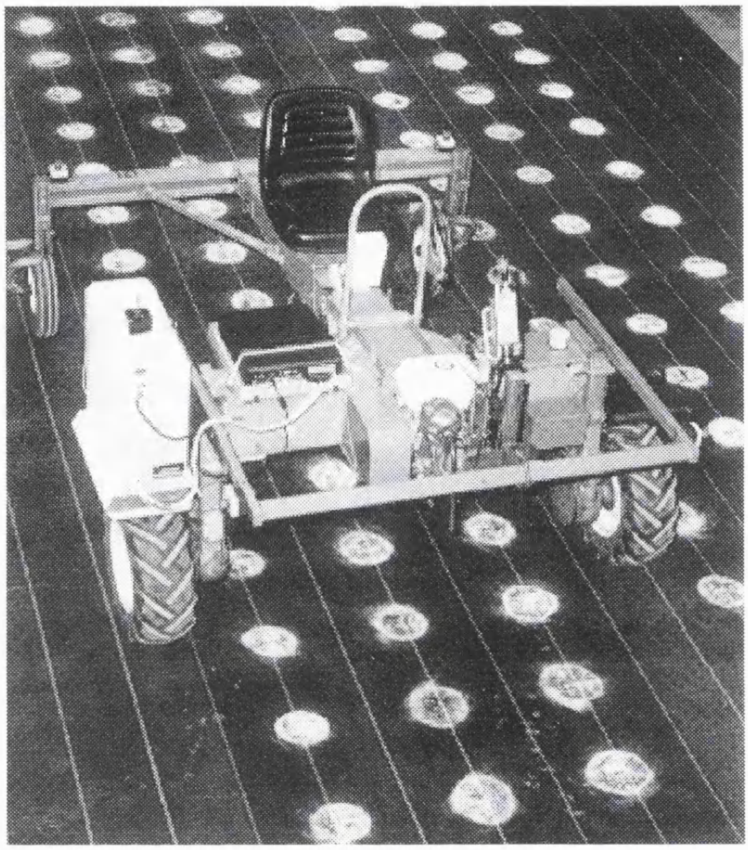


Figure 5.11: The vehicle in the test-bed environment.

The experimental procedure was to programme the vehicle to travel a distance of 12 metres (the length of the mat) along a trajectory composed of 1 m of acceleration, 10 m at a constant velocity (0.6 ms^{-1}) and 1 m deceleration to rest. The acceleration and constant velocity stages were performed with the vision subsystem switched on and being used to aid vehicle localisation as described in section 4.6, whilst the deceleration stage used dead-reckoning alone. During transit, the vehicle left a trail of sand along the mat whilst logging its estimated position into a file. At the end of the run, the position of the sand trail was measured with respect to the central row of circles at 0.25 m intervals. From an experimental measurement standpoint, the test-bed has a considerable advantage over real crop in that the position of the central row is well defined. In the field it is much more difficult to judge this central position.

Evaluation of the system's performance over a run is provided by studying the root mean square error (r.m.s.e.) between the measured points from the sand trail and the corresponding es-

timate logged by the vehicle controller. Such a metric was used by Hague and Tillett [HMT97a] to measure the system's performance with the Hough transform algorithm of Marchant and Brivot [MB95] in the field. A similar test-bed experiment was also carried out by Hague and Tillett [HT96] to assess the performance of the system with the Hough transform. An r.m.s.e figure was not given for this test, but it was stated that the error was typically in the range $\pm 15mm$.

A further metric is used in this thesis to analyse the bias of the estimator. This metric is the mean difference between each measurement taken from the trail and its corresponding estimate from the navigation system. If the estimator is unbiased, this mean difference should be zero.

It should be noted that the vehicle's position estimator does not directly gauge perpendicular offset of the vehicle from the central row. This distance h (as seen in figure 5.12) can be calculated using the estimated bearing angle Ψ and offset in camera co-ordinates t_x by the following formula

$$h = t_x \cos \Psi, \tag{5.1}$$

and it is this calculated value for h which is plotted in the results section below.

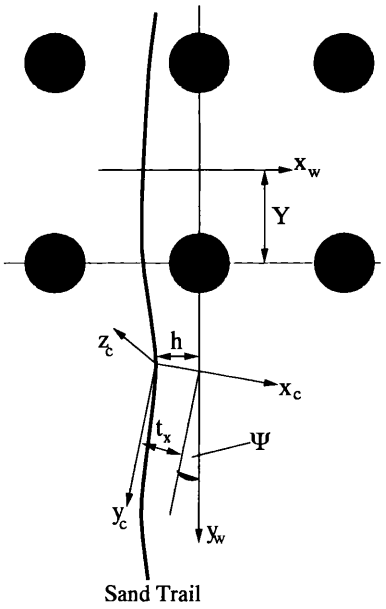


Figure 5.12: The perpendicular offset h.

5.2.1 Results

Six experimental runs were performed, and the estimated and measured positions are plotted in figure 5.13. In each figure the solid trace denotes the estimated position, and the dashed line the measured position. In addition to logging the estimated position, the filter also logs the uncertainty of the position estimate by recording a term from the filter covariance matrix. These

uncertainty measures are plotted in figure 5.14, and are discussed further below.

Before discussing the r.m.s. error figures, it is interesting to look at the shape of these standard deviation curves. When the vehicle starts up, the uncertainty is high; the automatic initialisation algorithm (presented in the next chapter) has yet to be implemented in the on-line system, and the filter is started with an approximate position estimate with high uncertainty, both chosen by hand. With the onset of motion, extra information is provided by the inclusion of the kinematic model which links together the estimates of parameters t_x and Ψ (equation 4.3). The uncertainty is thus driven down to an approximately steady level. During deceleration in the last metre of travel, the vision system is turned off and the uncertainty begins to rise again. This demonstrates the importance of the vision data to position estimation.

To counter the effects of the approximate initialisation, which would otherwise dominate the results, the r.m.s.e performance measure has been calculated only for the part of each track where the measured and logged tracks appear to have converged (from c. $2.5m$ onwards in each case). These error figures are presented in table 5.22 below, together with the the mean of the filter's standard deviation over the same distance. Comparison of the two figures shows that the

Track	r.m.s.e. (mm)	error estimate (mm)
(a)	6.18	3.13
(b)	7.12	3.27
(c)	5.84	3.12
(d)	6.29	3.19
(e)	7.01	3.25
(f)	7.28	3.15

Table 5.22: The root mean square error between the measured and estimated offsets, and filter error estimate for the six tracks of figure 5.13.

measured error is much greater than the estimated errors. Such over-confidence is a well known trait of the extended Kalman filter [May90], and in some cases can cause the estimate to diverge from the true state, so that tracking fails or estimates become biased. Perusal of the tracks in figure 5.13 shows that divergence has not occurred, and a mean error of only $0.65mm$ between each measurement-estimate pair indicates that the system bias is very small.

The forward distance estimate is also of interest. The vehicle was commanded to run a total length of 12 metres ($1m$ acceleration, $10m$ constant velocity, $1m$ deceleration to rest); table 5.23 shows the measured and estimated path lengths for each run. As can be seen in the table,

the best estimates are within one centimetre of the measured length (tracks (a) and (c)) and the largest error is $9cm$ (track (e)). Such errors, although larger than the typical disagreements between the human assessments and automatic results seen in the off-line experiments (tables 5.2 and 5.5), are quite acceptable for the task of navigating along the crop rows. Differences between the off-line and on-line figures may be attributable to many sources including systematic errors in human assessment of the off-line data, wheel slip on the mat (although this is expected to be small), measurement noise of the trail on the mat, or simply the fact that the on-line experiment operates over a $12m$ run, whereas the off-line sequences represent approximately $1.2m$, and error accumulates over distance.

Track	Measured Length (m)	Estimated Length (m)
(a)	12.04	12.05
(b)	11.98	12.06
(c)	12.07	12.06
(d)	12.03	12.07
(e)	11.97	12.06
(f)	11.97	12.05

Table 5.23: Estimated and measured track lengths.

5.3 Summary

Experiments have been designed to test the viability of the proposed crop grid tracking algorithms. The first test measures the consistency of the off-line algorithm's performance on image sequences. In the absence of ground truth information as to the vehicle's position, the tracked state trajectories are compared with estimates from the Hough transform algorithm of Marchant and Brivot, an algorithm known to provide satisfactory performance in the field [HMT97a], and with measurements by three human observers over the same sequences. The algorithm is tested with both hand-picked and automatically selected image features, and found to compare favourably with the Hough transform method in both cases. The method of feature selection is the subject of the following chapter. The advantages of estimating the grid parameters on-line is also shown, with improved agreement with human measurements when the grid model better reflects the underlying crop spacings.

Once we had gained confidence in the algorithms from the off-line tests, the next step was to perform an on-line experiment with the vehicle in a simplified test-bed environment by using a black mat painted with a regular grid of white circles to represent an idealised crop. This

second experiment allows comparison of the vehicle's navigation system estimator, which combines vision data and dead-reckoning information, with ground truth measurement of the vehicle's position in the test-bed environment. The system performance was found to be sufficiently accurate for navigation along the grid formed by the circles, and shows promise for navigation in a field of real plants. Experimental results assessing the navigation performance outdoors in the field environment are presented in chapter 8. So far, only the algorithm with fixed grid parameters (AUTO) has been tested on-line. Future work must involve an on-line implementation of the more sophisticated algorithm which also estimates the crop spacing parameters (AUTO2).

Thus far, we have presented the crop grid tracking algorithms and demonstrated their performance off-line on image sequences captured from the vehicle, and on-line in a simplified test-bed environment. The next chapter covers issues that are vital to successful tracking with extended Kalman filters. The first of these is the provision of an estimate of state and covariance to initialise the extended Kalman filter. The second is data association, which is required to select appropriate observations from the set of features generated by the image processing system.

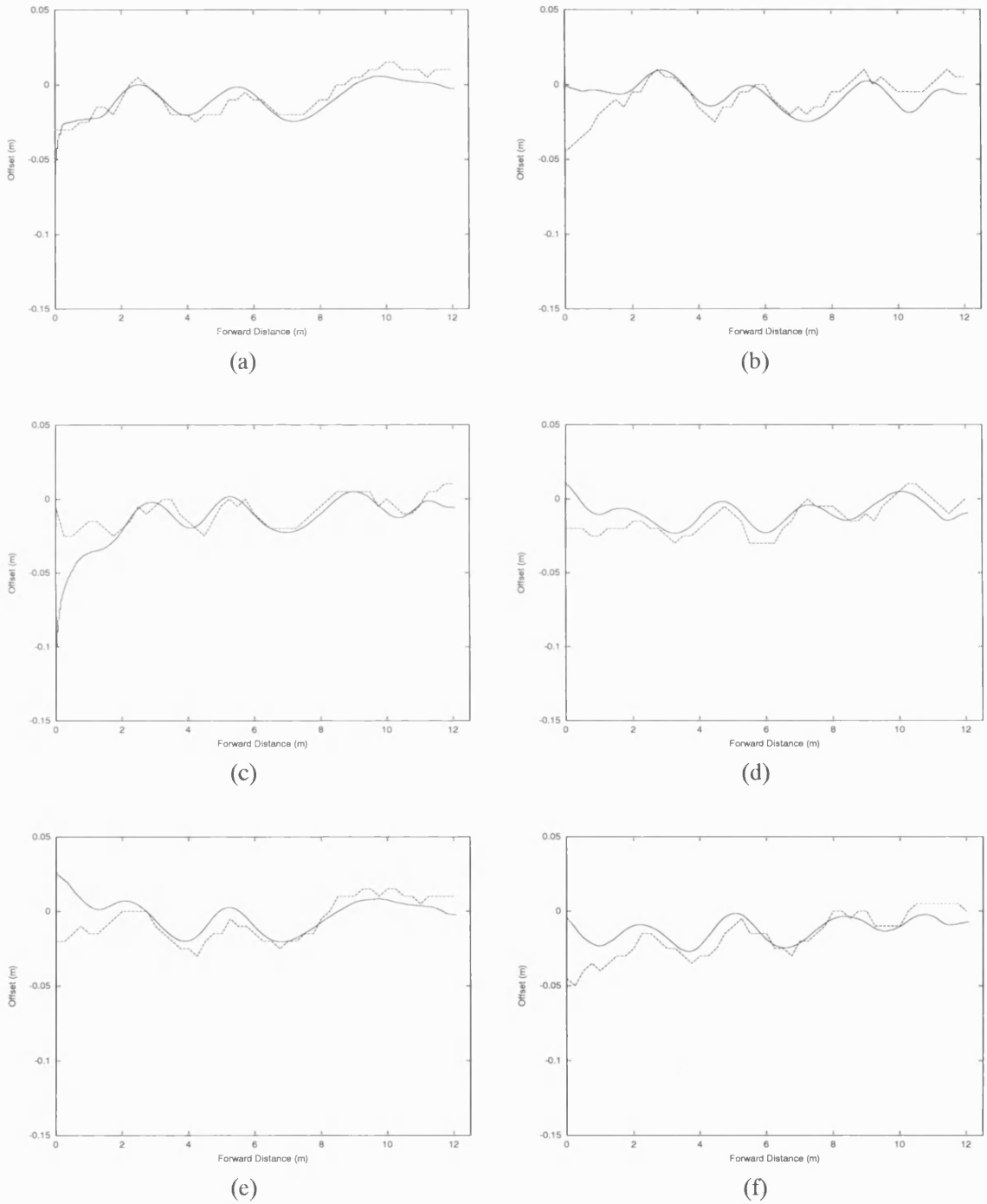


Figure 5.13: Navigation trials. Each figure (a)-(f) shows the estimated (solid) and measured (dashed) offset of the vehicle from the central row of crop.

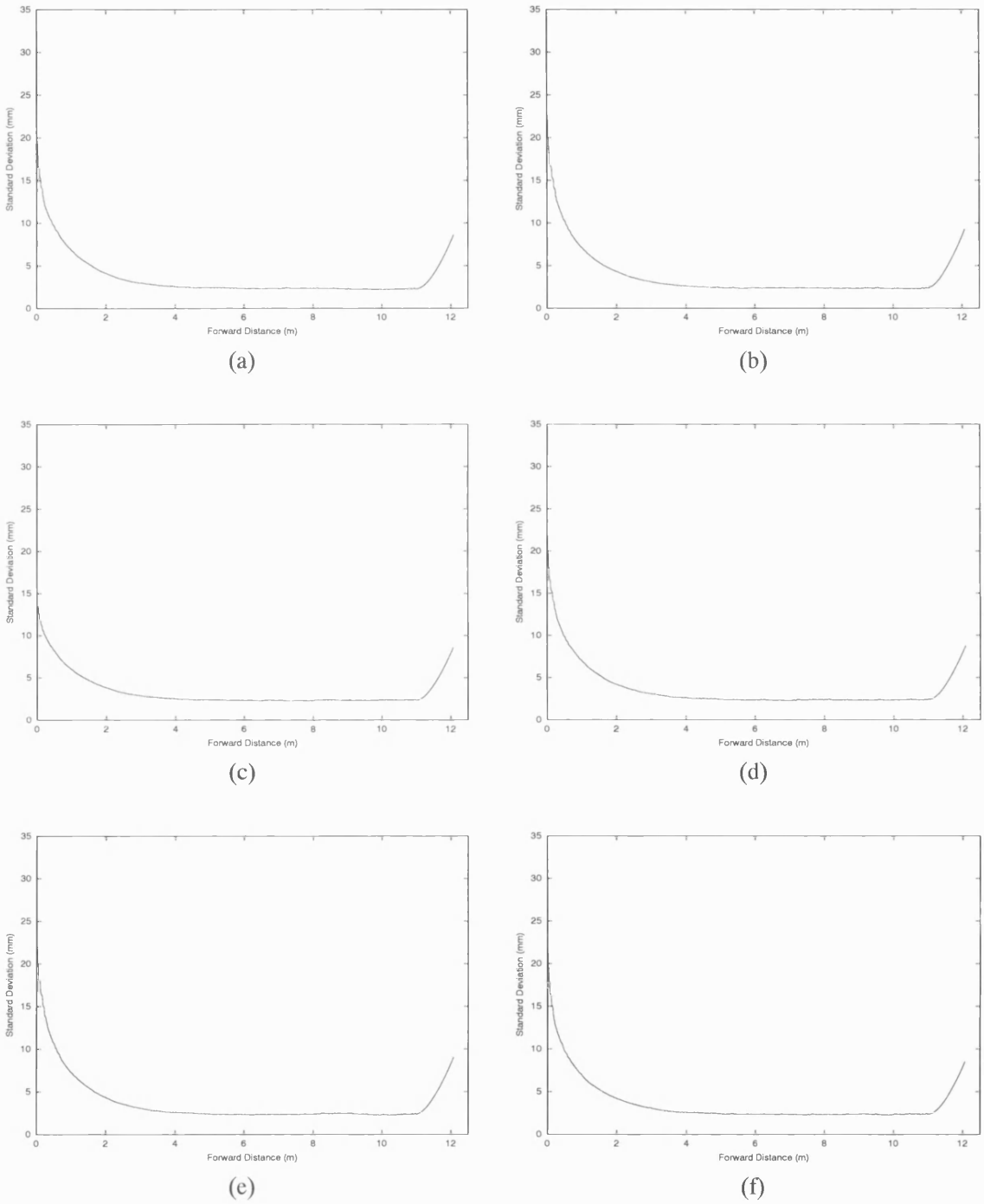


Figure 5.14: Estimate uncertainties. Each figure (a)-(f) shows the standard deviation (as given by the square root of the filter's estimate covariance matrix) of the position estimate in mm.

Chapter 6

Initialisation and data association

The development and testing of the crop grid tracking algorithm has been documented in the previous two chapters, and it shows great promise as a localisation tool for the autonomous horticultural vehicle. There are two issues that have received mention, but as of yet no explanation. These are the initialisation of the extended Kalman filter, and data association.

As is well known [BH97, BSF88], the extended Kalman filter algorithm is recursive, and refines its estimate of the state (and its covariance) by predicting the state's value in the next instant and then correcting this prediction using new measurements. This predict-correct cycle needs to be initialised with a starting estimate of the state and its covariance. In particular, initialisation can be especially important for the stable performance of extended Kalman filters, as was noted in chapter 3, in order to ensure an appropriate linearisation point for computation of the Kalman gain. To initialise the crop grid tracker, a two stage algorithm has been devised. The first stage uses the Hough transform method due to Marchant and Brivot [MB95] to locate the crop rows in the first image of the sequence, yielding an estimate of t_x and Ψ (see appendix A for details). This is followed by a second stage which obtains an initial estimate of Y via a novel use of the discrete Fourier transform to locate the offset of the bottom-most plant in the image. In the case of the crop grid tracker with grid parameter estimation, the initial values of $\bar{r}$ and $\bar{l}$ are given by hand. The covariance of the initial estimates on t_x, Y and Ψ is derived from an evaluation of the two-stage algorithm's performance. The variances of $\bar{r}$ and $\bar{l}$ are given by knowledge of the crop planting process, obtained from measurements taken in the field. Currently, this automatic initialisation has only been implemented in the off-line system. The on-line system is initialised by hand, with a "large" variance and is allowed to converge over a series of frames before the vehicle is started. This technique is surprisingly reliable, although the algorithm does fail to converge approximately 5% of the time, suggesting that the automatic method should be used.

The second issue tackled in this chapter is data association, the process of matching sensed

data (plant matter features extracted by image processing) with predictions (the predicted position of the crop plants in the crop grid) and is vital to the success of tracking. The Kalman filter state estimate update equation (equation 3.12 in chapter 3) assumes that there is a single observation $\mathbf{z}(k)$ which corresponds to the state prediction $\hat{\mathbf{x}}^-(k)$. In most real systems, this is not the case, and there are many possible matches to a single prediction. In the crop grid tracker, we require a single crop plant to be matched to a predicted crop grid position, but the image processing step produces both crop and weed features, so some method of pairing the predicted crop grid positions with suitable image features is required. A nearest neighbour approach is used to match a single image feature with each predicted crop position. A so-called "validation gate" is used as a pre-cursor to data association to reduce the number of potential matches and also to cater for instances where the plant corresponding to a grid position has failed to grow.

6.1 The initialisation algorithm

We are fortunate in our horticultural application, because the crop field environment is quite simple and only a single instance of the object of interest (the crop planting grid) is present in the field of view, and it is unlikely that any distractors will be present (i.e. a set of weeds lying in a formation that mimics the crop grid). The general problem of initialising object trackers in the presence of multiple instances and/or distractors is more difficult [Wil97], although recently a technique dubbed "Bayesian Correlation" [SBIM99] has been developed. Bayesian correlation uses learned models of the response of both object foreground and image background to a set of filters to construct an estimate of object position likelihood over an entire image. Moments of the distribution, such as mean and variance (required for Kalman filter initialisation) can be calculated directly from the likelihood function. For our purposes however, the more straightforward two-stage algorithm described below has been found adequate.

To start the crop grid tracking process, an initial estimate of the crop pattern position $\mathbf{x}$ is acquired in two stages. Firstly, the initial estimates of t_x and Ψ are obtained using the Hough transform method developed by Marchant and Brivot [MB95] (see appendix A). Secondly, the grid position Y is then obtained from Fourier analysis of 1D image samples taken along the extracted rows.

If we sample the image along the crop rows located by the Hough transform, a grey-level profile is obtained. Figure 6.1 shows an idealised binary image with a row marked and its corresponding sample, with figure 6.2 showing a real image and the sample taken from that image. The sampling is performed in world frame co-ordinates, which accounts for the regular spacing of the sample peaks in the figure, despite the perspective foreshortening in the image. To allow

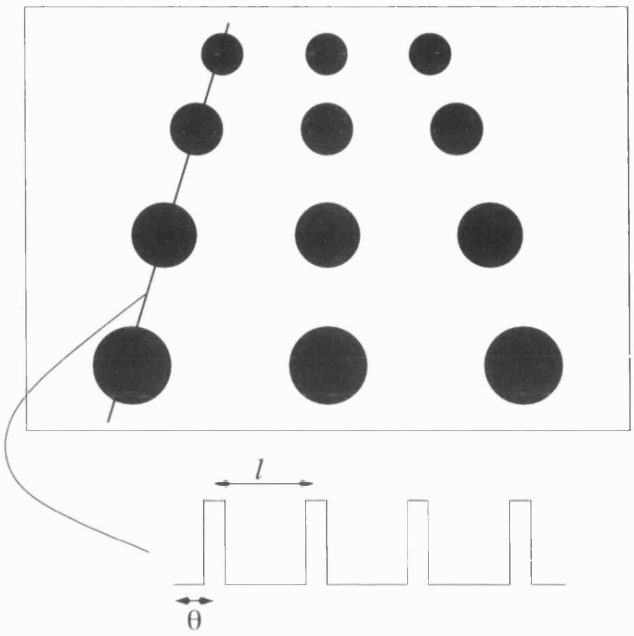


Figure 6.1: Sampling along a row: the phase θ of the sample provides the offset of the planting pattern.

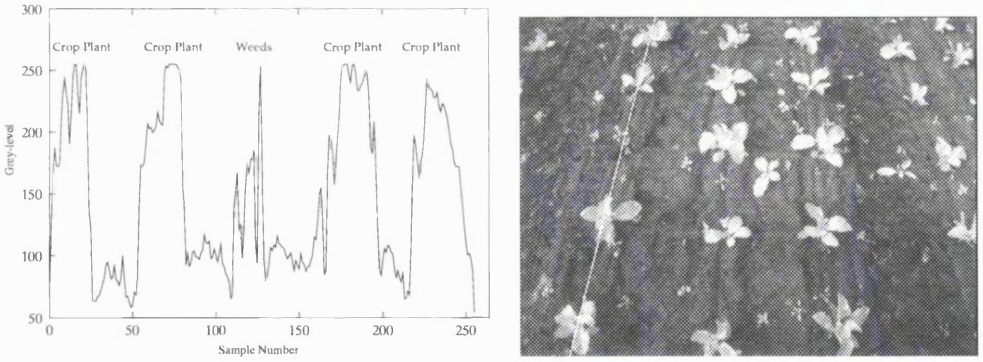


Figure 6.2: Left: a row sample from a typical image – the grey-level peaks corresponding to plant matter are hand-labelled crop or weed. Right: the image sampled the white line indicates the row sampled.

for non-ideal positioning of plants along the row, and the fact that the plants are two dimensional objects in the image rather than one dimensional, the sample to be analysed is constructed by taking the mean (at each sample point) of a set of 5 samples taken 5 mm apart on the ground plane around each row identified by the Hough transform method.

The procedure for obtaining the required offset values from these samples commences with the calculation of the discrete Fourier transform $F(j\omega)$ of the (1D) grey-level profile (ω is the

angular spatial frequency in radians per metre) and proceeds via calculation of the phase θ of the coefficients corresponding to the frequency $2\pi/\bar{l}$, the expected frequency of plant spacing $\bar{l}$ from the crop model. θ may then be converted into spatial offset along the row using the following formula

$$\text{Offset} = \frac{\theta \bar{l}}{2\pi}. \tag{6.1}$$

By using this method, model position Y can be calculated from the mean value of the offset measurement from each row.

Once initial values for t_x, Y and Ψ have been obtained, the state estimate $\hat{\mathbf{x}}^-(0)$ may be formed, leaving only the initial state covariance $\mathbf{P}^-(0)$ undetermined. Marchant and Brivot [MB95], estimate the root mean square error of the estimates obtained from their Hough transform algorithm. They give resultant r.m.s errors of 12.5 mm on t_x and 1^o on Ψ. As noted by Bar-Shalom and Fortmann [BSF88], in the extended Kalman filter the matrix $\mathbf{P}$ is not strictly a covariance, but a measure of mean square error on the estimate $\hat{\mathbf{x}}$, so these values of offset and angular error may be used directly.

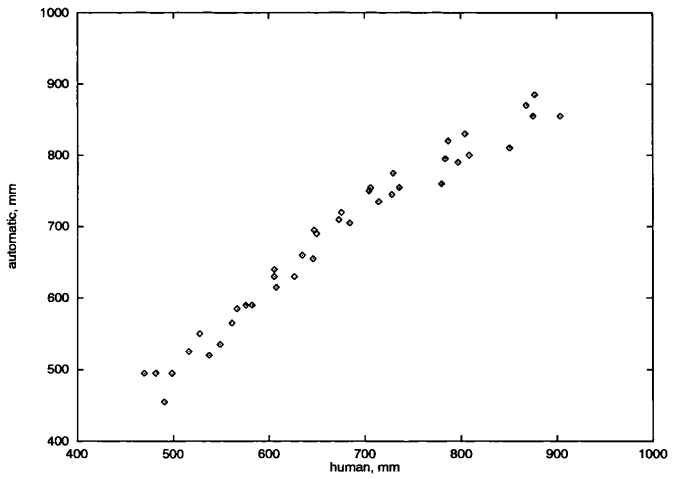


Figure 6.3: Comparison of human and automatic assessment of row offset. Note that the points lie approximately on a straight line at 45 ^{o}.

Finally, to obtain a measure of mean square error for the estimate of the row offset, and hence Y, the mean-square difference metric used as a similarity measure in chapter 5 has been employed. The Hough transform was used to provide the initial position of the row structure, and then a template was aligned by hand to determine the offset Y from each of 40 images (each image in sequence 1 and sequence 2 from chapter 5). Figure 6.3 plots the automatic measurements of Y against those derived from human measurements, and a mean-square difference measure of $24.5mm$ was calculated. As explained in chapter 5, this difference metric assumes equal distribution of errors between the algorithm and the human assessment. The difference measure

produced is not necessarily a mean-square error, unless the errors are in fact equally distributed between algorithm and human, but when veridical ground-truth measurements are unavailable, this kind of comparison with subjective human judgement provides a pragmatic solution to the problem of estimating the initial state covariance.

Experiments on image sequence 1 of the previous chapter in which the initial covariance was increased by up to a factor of 10, or decreased to zero, showed that only the initial response of the filter was affected , with convergence to the same track after two or three images into the trial sequence.

6.2 Data association and validation

The second issue of concern in this chapter is data association. When tracking an object in the presence of clutter (observed features which do not represent the object of interest), a data association policy is essential to link the observation process to the estimation mechanism [Rao92]. We are using the extended Kalman filter as the estimation mechanism for the crop grid tracker (chapters 3 and 4), and the observation process takes the form of the image thresholding and chain coding algorithms of chapter 2. We know from the Kalman filter update equation (equation 3.12 in chapter 3) that a single observation $\mathbf{z}(k)$ is required to match each prediction $\hat{\mathbf{x}}^-(k)$. Unfortunately, in many applications, including ours, the observation process produces a number of candidate features, one or many of which may correspond to the target of interest, and some of which may be clutter. Data association is the process of sifting through these candidate features and selecting those that represent the target being tracked. In our crop grid tracking application, the image processing algorithms extracts both crop plant and weed features from each image, so a data association strategy is required to match crop features with the corresponding predicted positions of plants in the crop grid.

The image processing described in chapter 2 produces a list of features characterised by their centroid position (in pixel co-ordinates) and size (measured as a number of pixels). For convenience of exposition, we shall assume that each image feature contains pixels that exclusively represent either a crop plant or a weed[1]. The centroid of each feature is a candidate observation $\mathbf{z}_d(k, m, n)$ to match the predicted position of the plant at crop grid position m, n at time k. This predicted plant position is denoted $\hat{\mathbf{z}}^-(k, m, n)$, and is defined by the expression

$$\hat{\mathbf{z}}^-(k, m, n) = \mathbf{h}(\hat{\mathbf{x}}^-(k), m, n), \tag{6.2}$$

which is familiar from equation 4.13, with the state vector prediction $\hat{\mathbf{x}}^-(k)$ substituted for

[1] In reality, the image processing is not perfect, so each feature may contain a mixture of pixels that belong to crop, weed or soil, which will affect the final image segmentation. This point will be revisited in chapter 7.

$t_x, Y, \Psi, \bar{r}$ and $\bar{l}$.

The data association policy produces a feature $\mathbf{z}(k,m,n)$ from the set of candidates $\mathbf{z}_d(k,m,n)$ that "best" matches the predicted feature position $\hat{\mathbf{z}}(k,m,n)$. We have used a nearest neighbour data association algorithm [BSF88]. The nearest neighbour approach selects a single feature from the set of candidate matches that produces the smallest "normalised innovation". What we mean by the "normalised innovation" is defined below. Obviously, if a crop plant has fractured into individual leaves, only one leaf will be selected using this method. Many alternative data association strategies have been proposed in the literature, but the nearest neighbour approach is adequate for our purpose. Rao [Rao92] presents a review of alternative association techniques which encompasses "all neighbour" methods (where a number of features are combined to provide a single estimate) and multiple-prediction, multiple-feature approaches.

To explain the nearest neighbour data association algorithm, we define the *innovation* $\nu_d(k,m,n)$ as

$$\nu_d(k,m,n) = \mathbf{z}_d(k,m,n) - \hat{\mathbf{z}}^-(k,m,n), \tag{6.3}$$

which, being the difference of two random variables, has covariance matrix

$$\mathbf{S}_d(k,m,n) = \mathbf{R}_d(k,m,n) + \mathbf{h}_{\hat{\mathbf{x}}^-(k)}(\hat{\mathbf{x}}^-(k),m,n)\mathbf{P}^-(k)\mathbf{h}^T_{\hat{\mathbf{x}}(k)}(\hat{\mathbf{x}}^-(k),m,n), \tag{6.4}$$

where $\mathbf{h}_{\hat{\mathbf{x}}^-(k)}(\hat{\mathbf{x}}^-(k),m,n)$ is the matrix of partial derivatives of the observation function $\mathbf{h}(\mathbf{x}(k),m,n)$ with respect to the state variables, and $\mathbf{P}^-(k)$ is the state prediction covariance matrix. $\mathbf{S}(k,m,n)$ is the sum of two terms which represent the uncertainty on the observed position of feature $\mathbf{z}_d(k,m,n)$ and the predicted position of feature m,n at time k respectively.

The *normalised innovation* is the Mahalanobis distance γ^2 between $\mathbf{z}_d(k,m,n)$ and $\hat{\mathbf{z}}^-(k,m,n)$:

$$\gamma_d^2 = \nu_d(k,m,n)^T\mathbf{S}_d^{-1}(k,m,n)\nu_d(k,m,n), \tag{6.5}$$

and the nearest neighbour policy chooses the feature $\mathbf{z}_d(k,m,n)$ which minimises γ_d^2, i.e.

$$\mathbf{z}(k,m,n) = \arg\min_d \left\{[\mathbf{z}_d(k,m,n) - \hat{\mathbf{z}}^-(k,m,n)]^T\mathbf{S}_d^{-1}(k,m,n)[\mathbf{z}_d(k,m,n) - \hat{\mathbf{z}}^-(k,m,n)]\right\}. \tag{6.6}$$

Thus, to find the nearest neighbouring feature, the quadratic sum of equation 6.5 is evaluated for each candidate feature, and the feature which produces the smallest normalised innovation is chosen as the best match. γ_d^2 reflects the likelihood that the chosen $\mathbf{z}_d(k,m,n)$ represents the same point in observation space as that predicted to be at $\hat{\mathbf{z}}^-(k,m,n)$. The assumption is that true target features (crop plants) are more likely to satisfy the nearest neighbour test than clutter (weeds).

6.2.1 Feature validation

The nearest-neighbour algorithm described above assumes that each predicted plant position has a single corresponding image feature. As noted above, in some cases, the image processing algorithms produce more than one feature per crop plant. The next chapter therefore presents an algorithm for clustering multiple features which may represent parts of the same plant. In other cases, however, a crop plant might fail to grow, and there will be no image features that correctly correspond to the predicted crop grid position. To deal with such instances, we use a *validation gate* [BSF88] to place a constraint on the position that a matched feature may take. Without validation the prediction would be matched with a weed feature, or even a crop plant which correctly corresponds to another point on the grid. Such erroneous matches would lead to inappropriate state updates, and possibly loss of track.

The validation gate places a limit on the value of normalised innovation γ_d^2 (equation 6.5) for a permitted match, i.e. only those features $\mathbf{z}_d(k, m, n)$ for which

$$\gamma_d^2 \leq \chi_T^2, \tag{6.7}$$

where χ_T^2 is a threshold limit, are passed to the nearest-neighbour algorithm (equation 6.6). The symbol χ_T^2 has been chosen deliberately, because the normalised innovation is the sum of squared normally distributed random variables (the projection of the state estimate into the observation space), and as such is drawn from a χ^2 distribution of degree n, where n is the dimension of $\mathbf{z}(k, m, n)$ [Spi80][2]. A prediction ($\times$) and an observation ($+$) are illustrated in figure 6.4, together with the validation region defined by equation 6.7, which is marked as an elliptic contour centred on the prediction.

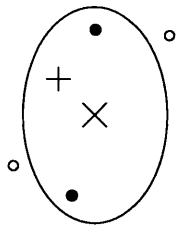


Figure 6.4: A validation gate. The prediction is marked with a '$\times$', the matched feature with a '+', some validated but unmatched features with '$\bullet$', and some features which fail the validation test with '$\circ$'.

If we recall that γ_d^2 is a also level of significance, then it can be seen that the validation gate (equation 6.7) will reject potential matches which are unlikely to be correct. Of course, if

[2]Strictly, in the extended Kalman filter, the state variables are generally not Gaussian random variables, but are approximated as such. This approximation is also made here.

a true match does not exist (e.g. if a crop plant has died), then enlarging the acceptance region increases the likelihood of an incorrect match (e.g. with a weed feature) being made. In practice, χ_T^2 is best set experimentally and we have found that $\chi_T^2 = 5.99$ allows for the imprecision in the position of individual crop plants relative to the grid whilst preventing inappropriate matches. This value of χ_T^2 corresponds to a 95% likelihood that the correct match will lie in the validation region. If no observations lie within the validation region, then the updated state estimate $\hat{\mathbf{x}}(k)$ is set to its predicted value $\hat{\mathbf{x}}^-(k)$ and the covariance $\mathbf{P}(k)$ to $\mathbf{P}^-(k)$, and the filter cycle continues in the usual manner.

6.2.2 Implications for filter implementation

Validation gating is a common strategy for outlier rejection in practical Kalman filtering applications [Rao92], and is often essential for stable estimation on real data. The price of this stability is that by rejecting all features outside the validation region, the validation gate truncates the Gaussian distributions that are assumed throughout the filtering process. This truncation will inevitably lead to the rejection of some correct matches, thereby introducing a bias into the filter's estimate.

The use of a validation gate also has a further implication for the implementation of the extended Kalman filter used to track the crop grid model that pertains to the order in which observations are incorporated into the state estimate. In the standard filter cycle of "predict-validate-associate-correct" , the position of a grid point (m, n) is predicted, an observation (from the set of validated features) is matched to this prediction using the nearest neighbour strategy, and this feature is used to update the state estimate. The next grid point position is then predicted, and so on until all grid points in the image have been exhausted. In this sequential cycle, it is possible that incorporating a crop feature that is at an extremity of a validation region may lead to subsequent crop features being rejected inappropriately (and hence update information will be lost). Figure 6.5 illustrates the problem. Two crop plants (labelled 1 and 2, marked with + signs) are shown, together with predicted positions from the filter ($\times$) and their validation regions (the contours). The solid lines correspond to the case where feature 1 is incorporated before feature 2, and the dotted lines the reverse. In both cases, the first feature to be incorporated into the filter's estimate is close to the edge of the validation region. Once this feature has been used to update the filter estimate, it leads to a prediction and validation region that excludes the other feature.

The solution to this problem is simple. Instead of adopting the usual sequential predict-validate-associate-update cycle for each point on the crop grid, we use a "parallel" validation and association strategy, where the complete set of grid point positions is predicted using the filter prediction at the time that the image is digitised from the camera (the estimate $\hat{\mathbf{x}}^-(k+1)_0$

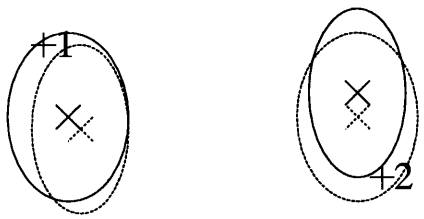


Figure 6.5: Order of data incorporation. Crop plants 1 and 2 are marked with + signs, whilst the × signs and contours denote the prediction positions and validation regions. Solid, feature 1 incorporated first, dashed feature 2 incorporated first.

as seen in section 4.3.3, chapter 4). Such a validation strategy is "natural" in the parallel update filter presented in chapter 4. This is illustrated in figure 6.6, where both plants 1 and 2 lie within the validation regions derived from the single prediction, so both observations will be matched to predictions and the update proceeds using these associations.

Figure 6.6: Parallel validation. A single state prediction is used to predict and validate both plant positions. In this case both features are validated.

6.3 Summary

Reliable operation of extended Kalman filters on real-world data often depends on two conditions. The first is that the filter should be provided with a good initial estimate of the state and its covariance matrix as tracking starts, and the second is the selection of appropriate measurements from the set of observations to update the state estimate.

We have presented a two-step algorithm to initialise the filter. The first step locates the crop row structure using the Hough transform developed by Marchant and Brivot [MB95], and then determines the grid offset via a novel use of the discrete Fourier transform. The crop grid parameters $\bar{r}$ and $\bar{l}$ are given by hand, as their mean values and variance are assumed to be known from the planting process. Initialisation requires that not only a state estimate is provided, but also the variance of this estimate. Our two-step algorithm does not provide the required variances directly, but we have been able to estimate them from evaluation of the algorithm's performance.

The image processing techniques described in chapter 2 produce a set of features from each image that describe plant matter, both crop and weed. The crop grid tracker predicts the position of the crop plants, but not the weeds, which appear as "clutter" in the set of observed features. Validation gating is used to reject image features that are unlikely to correspond to the predicted crop grid points, and a nearest neighbour data association policy selects single image features to match the grid points. In the previous chapter, the algorithms AUTO and AUTO2 used the nearest neighbour data association policy, and produced performance comparable with that of the SEMI and SEMI2 algorithms where features were matched with predictions by hand. However, since it is also possible that each crop plant may itself be represented by several features in an image (see chapter 2, section 2.3.8), a feature classification policy is required that takes this problem into account. This is discussed in detail in the next chapter.

Chapter 7

Image segmentation

So far, we have concentrated on the use of the crop grid tracking algorithm as an aid for navigation. Alternative crop grid models have been assessed, and an initialisation method proposed together with a data association policy to match image features to plants in the crop grid. However, the crop grid is not just useful as a landmark for navigation. Image features that lie "within" the grid may be assumed to represent crop plants, with the remainder representing weeds. Further knowledge of the agricultural domain may also be exploited to aid the segmentation process. The cauliflower crop used throughout this thesis are routinely grown into seedlings of c. 10 cm height in greenhouses prior to transplantation into the freshly tilled field. The crop, therefore, are well grown before the weeds can take root, and it can be safe to assume that the crop will generally be larger than the weed plants. Their size in the image is a useful cue for differentiating between crop and weed, as we noted in section 2.3.

The image processing stage (chapter 2) produces a set of blobs in the image which are characterised by their centroid (pixel co-ordinates) and size (in number of pixels). In this chapter, we present an algorithm for classifying these features as either crop or weed. The algorithm first filters the features on the basis of size, those smaller than a threshold size being classified as weed. The features above the size threshold are passed on to a clustering algorithm somewhat similar to the validation gate presented in the previous chapter. The clustering algorithm groups together features that lie close to the latest estimate of the crop grid position and classifies them as crop. The features that are not grouped by the algorithm are classified as weed.

The performance of the segmentation algorithm is analysed on ground truth plant matter data to determine the effectiveness of the algorithm independently of the plant matter/soil discrimination method, before being demonstrated in off-line experiments on the test image sequences A–D that were introduced in chapter 2. The size filtering and data association steps are controlled by threshold parameters, and methods of determining suitable operating values of these parameters, based upon ROC curve analysis, are discussed.

7.1 The image segmentation algorithm

The purpose of the image segmentation algorithm is to allocate each pixel to a class, be it crop plant (denoted C), weed (W) or soil (S). The algorithm performs this task in three stages. The first stage is the separation of the image into soil pixels and plant matter blobs. This is done by the adaptive interpolating threshold and chain-coding algorithm described in chapter 2, that extracts a set of blobs from the image which represent the plant matter. It is known that some soil pixels are incorrectly classified as plant matter and vice-versa. Each blob is described by its centroid (in pixel co-ordinates) and size (the number of pixels in the blob). These blobs make up the set of candidate plant matter features, denoted P, which are passed on to the next two stages of the algorithm. The pixels rejected by the threshold are classified as soil (S).

The next stage of the algorithm exploits the fact that the crop plants are grown to seedlings before being transplanted into the freshly tilled field. The crop plants are thus already an appreciable size before the weeds start to grow, and it is reasonable to assume that this size advantage is sustained if the crop is regularly tended. The plant matter features P are therefore filtered on the basis of size with those below the size threshold classified as weed (W), and those above it passed on to the final stage of the classification process.

The third stage of the segmentation algorithm utilises the tracker's estimate of the crop grid position and a clustering technique similar to the validation gate presented in chapter 6 to label features close to the crop grid structure as crop (C), and the remainder as weed (W).

7.1.1 Size filtering

Section 2.3 described the extraction of plant matter features from the image sequences by application of a grey-level threshold and clustering of neighbouring pixels using a chain-coding algorithm. Each cluster is described by its centroid and size. If the size of each blob reflects the size of the plant being imaged, then imposing a threshold on the blob size should be an efficient way of screening out the smaller weeds. It should be stressed that the aim of the size filtering steps is to sort the features into two sets; those which are most likely weeds, and others which might be weed or crop.

Figure 7.1 shows histograms of the size of weed and crop plant image blobs in the ground truth images from sequences A – D (chapter 2). It can be seen from the histograms that the vast majority (in fact 95%) of the weed blobs have a size of less than 50 pixels, whilst most (90%) of the crop blobs have a size of 50 or pixels or greater. This supports the claim that the weed plants are typically smaller than the crop. The blob sizes plotted in the histograms come directly from the perspective images captured from the vehicle's camera. The perspective projection will

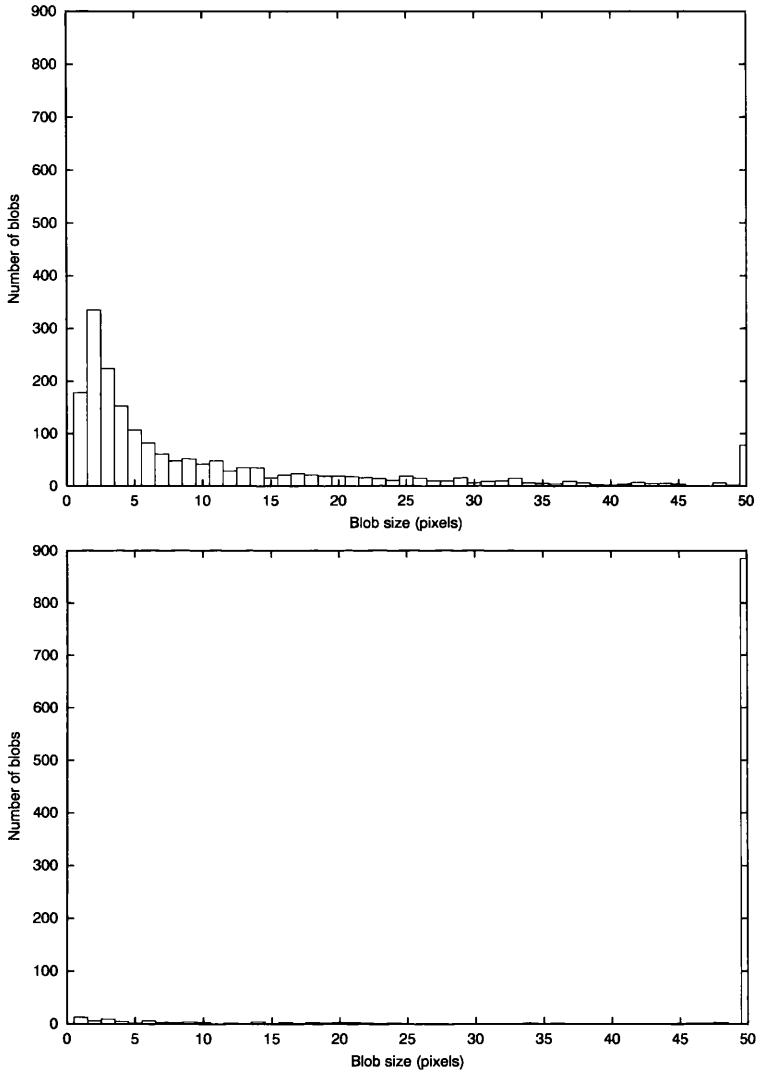


Figure 7.1: Blob size histograms. Top: weed blobs. Bottom: crop blobs. In both histograms, the right-most bin (n arked 50) counts all blobs of size ≥ 50.

make an object close to the vehicle appear larger in the image than a same-sized object further away. However, there is little overlap between the crop and weed size distributions, so we do not try to correct for such perspective effects.

Thus, we have a straightforward algorithm that places a threshold on the size s of the image features. This may be expressed as follows:

$$\text{Class(feature)} = \begin{cases} W & \text{if} \quad s(\text{feature}) < \varsigma \\ P & \text{if} \quad s(\text{feature}) \geq \varsigma \end{cases}, \tag{7.1}$$

where $s(\text{feature})$ is the size of an image feature in pixels, and ς is the size threshold.

7.1.2 Feature clustering

The size filtering algorithm has provided a set of candidate crop features, which retain the plant matter label P. These features are next sorted into crop (C) and weed (W) features on the basis of their position on the ground plane. Crop features should be close to the plant positions given by the crop grid model, whilst weeds will be more randomly scattered across the field. In chapter 2, it was seen that crop plants can fracture into multiple image features. For these features to be classified correctly, they must all be clustered together and associated with the crop plant positions in the crop grid. We perform the clustering with a heuristic algorithm akin to the feature validation process given in the previous chapter.

In chapter 6, we defined the validation gate, a region centred on the *predicted* crop plant position $\mathbf{h}(\hat{\mathbf{x}}^{-}(k), m, n)$ outside of which observations were not associated with the crop grid point (m, n). Here, we define an association region around the *updated* estimate of each crop plant position $\mathbf{h}(\hat{\mathbf{x}}(k), m, n)$, inside of which all observations are classified as crop plant, and assigned label C.

If we denote the position in the image of estimated crop grid location m, n as $\hat{\mathbf{z}}(k, m, n)$, we have the observation equation (equation 4.12),

$$\hat{\mathbf{z}}(k, m, n) = \mathbf{h}(\hat{\mathbf{x}}(k), m, n). \tag{7.2}$$

Now, if we have the set of observed candidate crop features P, each at an image position denoted $\mathbf{z}_p(k)$ (where $p \in P$), we can set up a clustering criterion, where $\mathbf{z}_p(k)$ is classified as crop if

$$[\mathbf{z}_p(k, m, n) - \hat{\mathbf{z}}(k, m, n)]^T \mathbf{S}_{assoc}^{-1}(k, m, n)[\mathbf{z}_p(k, m, n) - \hat{\mathbf{z}}(k, m, n)] \leq \chi^2_{assoc}, \tag{7.3}$$

where $\mathbf{S}_{assoc}$, described in detail below, is a covariance matrix that determines the shape of the association region, and χ^2_{assoc} controls the size of the association region. As with the validation gate in the previous chapter, we set $\chi^2_{assoc} = 5.99$ to reflect a 95% chance of finding all of the

crop features within the association region. Equation 7.3 has the same form as the validation gate seen in the previous chapter (equations 6.5 and 6.7), but with the corrected state estimate $\hat{\mathbf{x}}(k)$ in place of the predicted estimate $\hat{\mathbf{x}}^{-}(k)$.

The matrix $\mathbf{S}_{assoc}(k, m, n)$ describes association region centred on crop grid position m, n, and is composed thus

$$\mathbf{S}_{assoc}(k, m, n) = \mathbf{h}_{\hat{\mathbf{x}}(k)}(\hat{\mathbf{x}}(k), m, n)\mathbf{P}(k)\mathbf{h}^{T}_{\hat{\mathbf{x}}(k)}(\hat{\mathbf{x}}(k), m, n) + \mathbf{R}_{assoc}(k, m, n). \quad (7.4)$$

The first term on the right-hand side of equation 7.4, contains $\mathbf{h}_{\hat{\mathbf{x}}(k)}(\hat{\mathbf{x}}(k), m, n)$, the matrix of partial derivatives of the observation function $\mathbf{h}(\cdot)$ with respect to the state variables $\mathbf{x}(k)$ evaluated at the latest state estimate $\hat{\mathbf{x}}(k)$. $\mathbf{P}(k)$ is the latest state estimate covariance, so this first term describes the uncertainty in the estimate of the position of grid position m, n. The second term on the right-hand side of equation 7.4, $\mathbf{R}_{assoc}$, is a matrix that describes the association region constructed about that uncertain position, and we will return to it shortly. The construction of $\mathbf{S}_{assoc}$ is illustrated in figure 7.2.

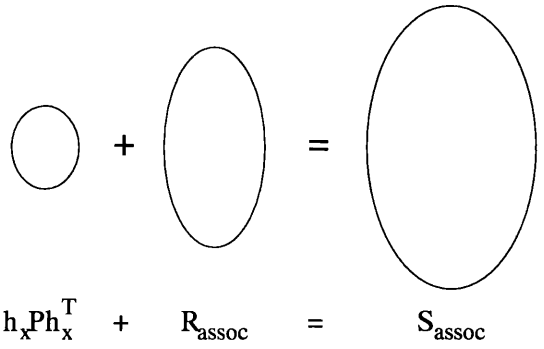


Figure 7.2: The construction of the clustering region $\mathbf{S}_{assoc}$.

All that is required now to fully determine $\mathbf{S}_{assoc}$ is specification of the matrix $\mathbf{R}_{assoc}$. In section 4.2.5, we set up the observation covariance matrix $\mathbf{R}_w$ on the ground plane, and used error propagation techniques to project it into the image to obtain the observation noise covariance matrix $\mathbf{R}(k, m, n)$ (equation 4.28). We follow a similar strategy here, where we construct a region in ground plane co-ordinates, described by a matrix $\mathbf{R}_r$, and then project the matrix into the image using error propagation techniques.

Suppose that we have perfect knowledge of a crop grid plant position on the ground plane, i.e. that $\mathbf{P}(k)$ in equation 7.4 is the zero matrix, and that this position is denoted $\mathbf{x}_{wc} = (x_{wc}, y_{wc})$, and that we have a feature in the image, $\mathbf{z}_p$, whose pixel co-ordinates (x_{fp}, y_{fp}) map onto the ground plane position $\mathbf{x}_{wp} = (x_{wp}, y_{wp})$. We will classify the feature $\mathbf{z}_p$ as crop if $\mathbf{x}_{wp}$

is within a radius r of the grid position $\mathbf{x}_{wc}$, i.e. if $\mathbf{x}_{wc}$ and $\mathbf{x}_{wp}$ satisfy the relationship

$$[\mathbf{x}_{wp} - \mathbf{x}_{wc}]^T \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} [\mathbf{x}_{wp} - \mathbf{x}_{wc}] \leq r^2. \tag{7.5}$$

Now suppose that

$$r = a\chi_{assoc}, \tag{7.6}$$

which allows us to write

$$[\mathbf{x}_{wp} - \mathbf{x}_{wc}]^T \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} [\mathbf{x}_{wp} - \mathbf{x}_{wc}] \leq a^2\chi^2_{assoc}, \tag{7.7}$$

or

$$[\mathbf{x}_{wp} - \mathbf{x}_{wc}]^T \begin{bmatrix} a^{-2} & 0 \\ 0 & a^{-2} \end{bmatrix} [\mathbf{x}_{wp} - \mathbf{x}_{wc}] \leq \chi^2_{assoc}. \tag{7.8}$$

Equation 7.8 is equivalent to interpreting the position of crop plant matter as a Gaussian distributed random variable with mean $\mathbf{x}_{wc}$ and covariance matrix

$$\mathbf{R}_r = \begin{bmatrix} a^2 & 0 \\ 0 & a^2 \end{bmatrix}, \tag{7.9}$$

and stating that all matter related to that crop plant lies within a radius r of $\mathbf{x}_{cw}$ with a statistical significance set by the value of χ^2_{assoc}. For any fixed values of r and χ_{assoc}, a is calculated from equation 7.6. This interpretation may seem curious, especially as the matrix $\mathbf{R}_r$ almost certainly does not reflect the distribution of plant features around the plant centre[1], but it allows straightforward calculation of the matrix $\mathbf{R}_{assoc}$ by projection of $\mathbf{R}_r$ into the image plane as follows:

$$\mathbf{R}_{assoc} = \mathbf{F}_w(x_{wc}, y_{wc})\mathbf{R}_r\mathbf{F}_w^T(x_{wc}, y_{wc}), \tag{7.10}$$

where the matrix of partial derivatives $\mathbf{F}_w(x_{wc}, y_{wc})$ is given in equation 4.29 on page 85.

All that is required to specify $\mathbf{R}_{assoc}$, and hence the association region $\mathbf{S}_{assoc}$, is a value for the parameter r. If r is too small, then crop features will lie outside of the association region and will be misclassified as weed. However, if r is too large, then weed features will lie inside the association region and be misclassified as crop. We will return to parameter value selection in section 7.2 below.

[1] Fracturing is caused by shadows falling on the plant, so the true distribution of features has a dependency on light intensity and direction, the shape of the plant and environmental features that cast shadows; in short it will be difficult to model.

7.1.3 Classification summary

The classification process, from thresholding to data association is summarised in figure 7.3 which illustrates the decisions and possible outcomes for each image pixel. On the left is the input pixel which is subjected to a number of binary tests, which either reject the pixel as a "negative" (R) or accept it as a "positive" (A). The final outcomes of the decision process is on the right of the diagram; the pixel is classified as either C (crop), W (weed) or S (soil).

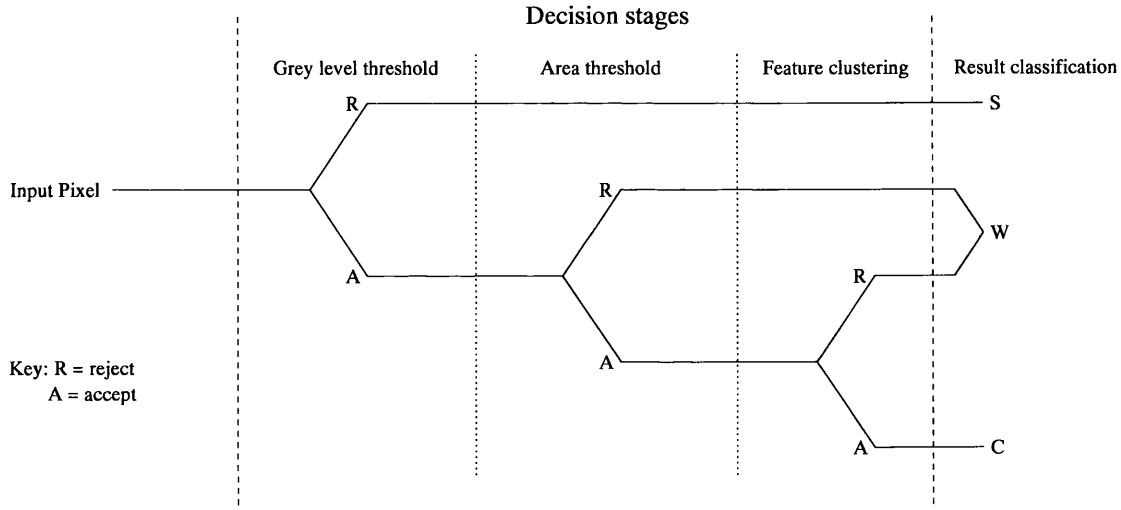


Figure 7.3: Segmentation schematic.

7.2 Operating point selection

In figure 7.3, the three decision stages of the image segmentation algorithm can be seen, and each stage is controlled by a threshold parameter. For the grey-level thresholding algorithm, this parameter is the threshold gain α and was set in chapter 2. The size threshold algorithm is controlled by a parameter ς, and the clustering algorithm by the radius r. Some method of determining an operating point for the crop/weed discrimination algorithm is required, i.e. suitable values for ς and r that will deliver "good" performance on the test image sequences A–D that were introduced in chapter 2.

In chapter 2, we used the receiver operating characteristic (ROC) curve to compare the performance of two grey-level thresholding algorithms and to determine a suitable operating point for the adaptive interpolating threshold algorithm. In that case, the algorithm had a single parameter, α, which was varied systematically to produce the points on the ROC curve. In the crop/weed discrimination algorithm we have two thresholds to set, ς and r. It is possible to adapt ROC analysis for the case where an algorithm has n thresholds (our algorithm has $n = 2$), as

we shall see below.

7.2.1 The maximum realisable ROC curve

A point in ROC space represents the performance of a particular classifier in terms of its true positive ratio (TPR, equation 2.12) and its false positive ratio (FPR, equation 2.13). A classifier is defined here as a particular instantiation of a number of algorithms, for example the combined size filtering and feature clustering algorithm with parameter settings $\varsigma = 40$pixels and $r = 0.30m$. Given any two classifiers, a and b, that are characterised by (fpr_a, tpr_a) and (fpr_b, tpr_b) respectively, and are joined by the straight line L_{ab} in ROC space (figure 7.4), Green and Swets [GS66] (and more recently Scott *et al* [SNP98]) show that a new classifier c, with mean performance (fpr_c, tpr_c) that lies on L_{ab} may be realised by randomly switching between the output of classifiers a and b. The probability of using the output of classifier a for the i^{th} decision is denoted $Pr(c_i = a)$, and is given by

$$Pr(c_i = a) = \frac{fpr_c - fpr_a}{fpr_b - fpr_a}, \tag{7.11}$$

and the probability that the output for decision i will be provided by classifier b is simply

$$Pr(c_i = b) = 1 - Pr(c_i = a). \tag{7.12}$$

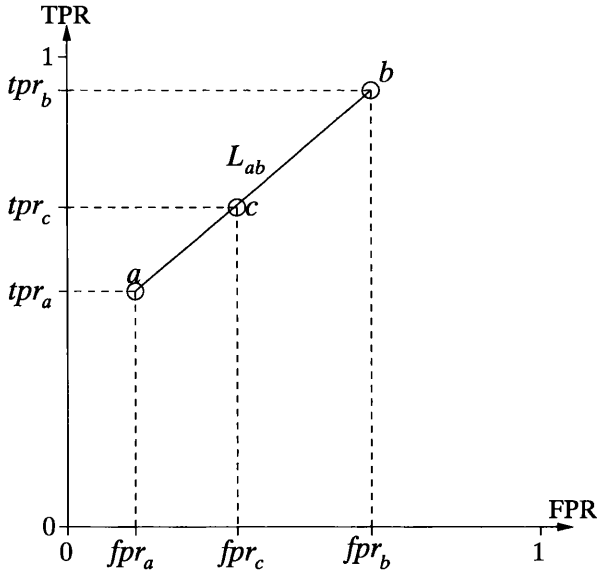


Figure 7.4: Two classifiers and a line in ROC space. Classifiers a and b are joined by a straight line L_{ab}. Any classifier c that lies on L_{ab} may be realised by randomly choosing between the output of classifiers a and b.

Equations 7.11 and 7.12 have been introduced to illustrate that any point on a straight line in ROC space that connects two existing classifiers is itself a classifier, and that a rule exists to

realise this classifier. This idea leads directly to the maximum realisable ROC (MRROC) curve [SNP98]. In the case where we have an algorithm controlled by a single parameter, each setting of the parameter results in a classifier that lies on a single curve in ROC space, the ROC curve. When we have n parameters to set, where $n \geq 2$, each different set of parameter values results in a different classifier, but these classifiers are no longer guaranteed to lie on a single curve in ROC space, but will populate the ROC space above the "chance diagonal" from (0,0) to (1,1).

If we have a set of existing n parameter classifiers plotted as points in ROC space, then their convex hull constitutes the maximum realisable ROC curve. The MRROC curve is said to be *maximum* because it encloses the largest area possible for any curve plotted through the points in ROC space. The MRROC curve is *realisable* because it is composed of a set of line segments that connect existing classifiers, so any point on the curve represents a realisable classifier. A portion of ROC space containing a group of classifiers and a section of the MRROC curve is plotted in figure 7.5. The area property of the MRROC curve are interpreted in exactly the same way as that of the ROC curves presented in chapter 2, however the MRROC is not differentiable in the same way as the usual ROC, so the slope does not have the same meaning.

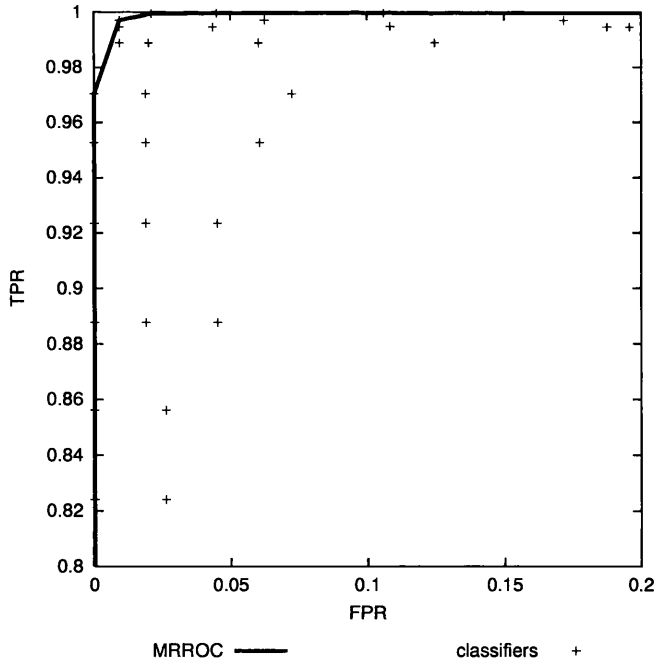


Figure 7.5: A maximum realisable ROC curve.

7.2.2 Parameter selection

To select suitable values of the algorithm parameters ς and r to segment each of the example sequences A–D introduced in chapter 2, we once again make use of the subset of ground truth images from each sequence where every pixel has been labelled, by hand, as either crop, weed,

soil or as a border pixel which is ignored for classification purposes[2]. For each ground truth image, the crop grid position is determined by hand, using the mouse-driven program that enabled production of the various HUMAN navigation data sets in chapter 5.

Now that we have a ground truth set of crop and weed features for a set of images, and the position of the crop grid in each of these images, we can form the MRROC curve for the set of classifiers produced by systematic variation of the size threshold ς and the clustering radius r. We assume that the human assessment of grid position is perfect, so the matrix $\mathbf{S}_{assoc}$ in equation 7.4 is simply given by $\mathbf{R}_{assoc}(k, m, n)$. Sections of the top left-hand corners of the MRROC curves produced for sequences A–D are plotted in figure 7.6, where crop pixels are the true positives and weed pixels true negatives.

The area under each MRROC curve is given in table 7.1. The area under the MRROC curve is close to the ideal figure of 1 for every sequence, so it would appear that the size filtering and feature clustering algorithms are effective for crop/weed discrimination. To set the operating

Sequence	Area under curve
A	0.9974
B	0.9997
C	0.9996
D	0.9993

Table 7.1: The area under the MRROC curve for the crop/weed discrimination in each ground truth image, sequences A – D.

values of ς and r, we choose the point on the MRROC nearest (in the Euclidean sense) to the $(0, 1)$ point, simply to achieve a blend of high true positive ratio and low false positive ratio. The operating points selected in this manner are given for each sequence in table 7.2. The area threshold ς levels determined for sequences A and C are larger than those for B and D, which could be expected because the crop plants are at a more advanced stage of growth in A and C. The radius threshold r of $450mm$ selected for sequence A is of similar size to the mean grid spacing parameters $\bar{r}$ and $\bar{l}$, so the algorithm is effectively segmenting the features on the basis of blob size alone. This reflects the fact that sequence A has very large crop plants and only a few, smaller, weeds. The true positive and false positive ratios associated with the operating points given in table 7.2 are shown in table 7.3, where positives refer to crop pixels and negatives to weed pixels.

[2]See section 2.3.4 for a discussion of why we exclude border pixels from the segmentation results.

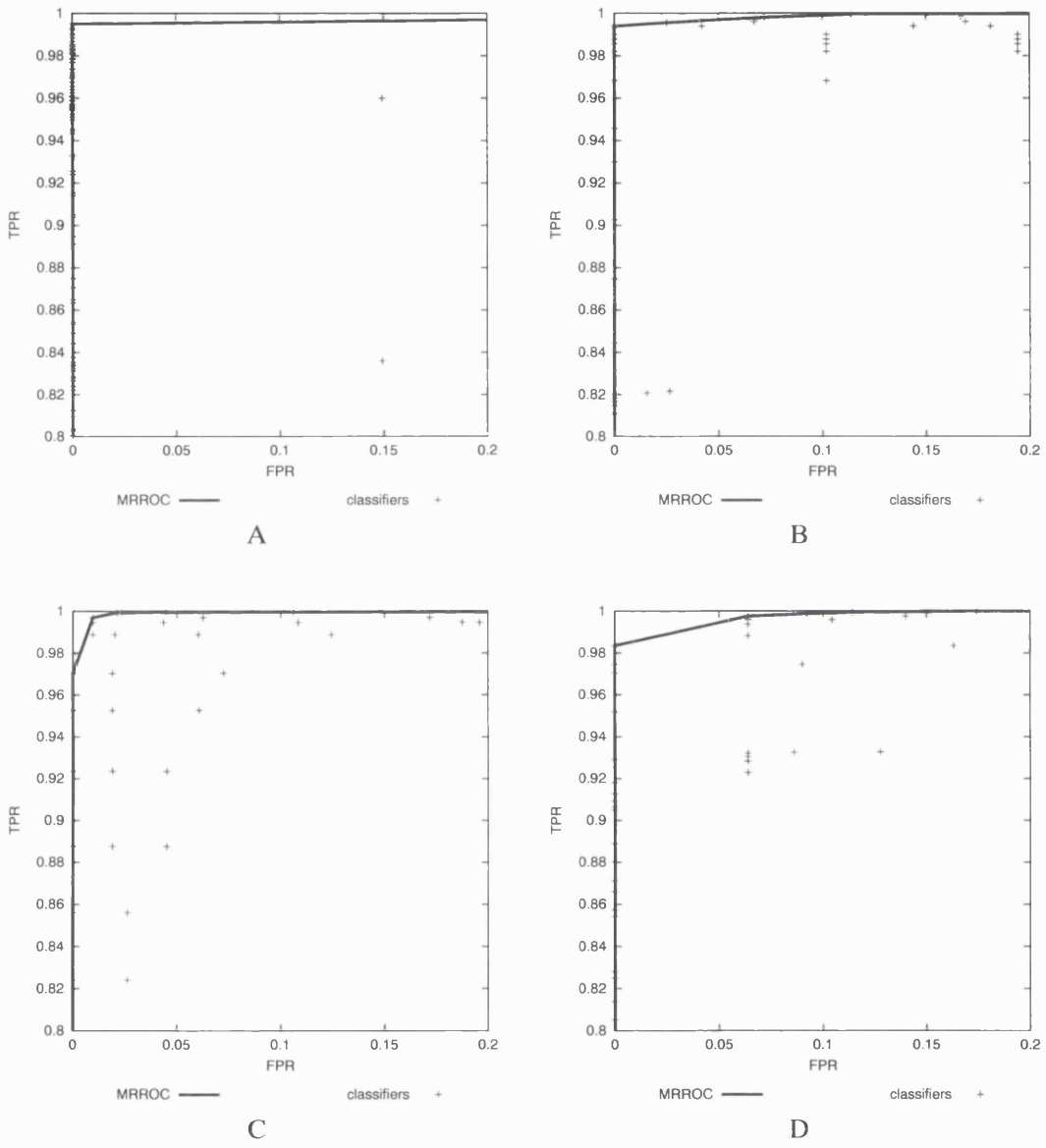


Figure 7.6: MRROC curves for ground truth plant matter segmentations of sequences A–D.

Sequence	ς (pixels)	r (mm)
A	100	450
B	30	100
C	80	100
D	30	100

Table 7.2: Operating points for the size filtering and clustering algorithms.

Sequence	TPR	FPR
A	0.9950	0.0
B	0.9940	0.0
C	0.9970	0.0094
D	0.9975	0.0638

Table 7.3: TPR and FPR for the operating points selected for sequences A – D.

A more principled selection of operating parameters might be possible if the values and costs of correct and incorrect decisions were known. For example, if the farmer wishes to remove all weeds and is willing to risk some crop in this process, the value of true negatives (correctly classified weed) would be high, and the cost of a false positive (weed classified as crop) would be higher than the cost of a false negative (crop classified as weed). If crop fertilisation was a priority, a true positive (correctly identified crop) would be high, and the cost of a false negative would be higher than the cost of a false positive.

7.3 Segmentation experiments

The size filtering and feature clustering algorithms have been tested in two off-line experiments on the four image sequences A – D. In both experiments, the crop grid tracking algorithm AUTO2 (see chapter 5), was used to estimate the crop grid position and grid parameters $\bar{r}$ and $\bar{l}$ throughout each of the four sequences. For each image where a ground truth crop, weed and soil segmentation was available, the estimate $\hat{\mathbf{x}}(k)$ of the grid position and its parameters, and the covariance $\mathbf{P}(k)$ for the estimate were stored.

The first experiment tests the ability of the size filtering and feature clustering algorithms to classify the ground truth plant matter images from each sequence when combined with the crop grid tracker. To some extent this experiment isolates the size filter and feature clustering from the effects of image processing errors. The second experiment applies the size filtering and feature clustering algorithms to the plant matter images produced by the image processing system. In this experiment, the full segmentation process described in figure 7.3 is under test.

7.3.1 Segmentation of ground truth plant matter features

The first experiment is designed as a test of the performance of the size filtering and feature clustering algorithms on the ground truth plant matter images, that, as indicated above, is as far as possible independent of the low-level image thresholding algorithm described in chapter 2. The test is not entirely independent of the image processing errors, because they have an effect

on the tracker's estimate of the crop grid position that is used in the feature clustering algorithm, but it does allow us to compare the true positive and false positive ratios for crop pixels directly with those given in table 7.3 in the parameter selection experiments.

For each of our ground truth images, we have corresponding estimates of the crop grid state $\hat{\mathbf{x}}(k)$ and its covariance $\mathbf{P}(k)$, provided by the crop grid tracker. The crop and weed features are extracted from the ground truth plant matter image and then size filtered (equation 7.1), before being passed on to the feature clustering algorithm (equation 7.3), which takes into account the uncertainty $\mathbf{P}(k)$ on the grid position $\hat{\mathbf{x}}(k)$. The size threshold ς and clustering radius r are given for each image sequence in table 7.2.

After processing, we have two sets of classified pixels for each image sequence. The first set is C, the ground truth plant matter pixels that have been classified as crop. The second set is W, the ground truth plant matter pixels that have been classified as weed. Given the two sets C and W, we can produce true positive (ground truth crop pixels that are classified C) and false positive (ground truth weed pixels classified as C) ratios for the automatic segmentation. These ratios are given in table 7.4 in the column marked 'automatic tracking'. The figures from table 7.3, which give the TPR and FPR for each sequence from the parameter setting experiment above are repeated for ease of reference in table 7.4 under the heading 'Parameter setting'.

	Automatic tracking		Parameter setting	
Sequence	TPR	FPR	TPR	FPR
A	0.9939	0.1564	0.9950	0.0
B	0.9982	0.0	0.9940	0.0
C	0.9981	0.0307	0.9970	0.0094
D	0.9993	0.2017	0.9975	0.0638

Table 7.4: TPR and FPR figures for the true plant matter images segmented whilst tracking, compared with those derived from the parameter setting experiment, for sequences A – D.

Before we compare the ratios from the tracking experiment with those from the parameter setting experiment, we should consider the main differences between the two experiments. In the tracking experiment, the association region (equation 7.3), within which all features are classified as crop, includes the uncertainty on the grid position, so will be larger than the corresponding region in the parameter setting experiment where the grid position was assumed to be known perfectly. We might expect that, as the association region expands, more image features

will fall within it, so both TPR and FPR are likely to rise. The second difference is that the grid position in the tracking experiment is determined automatically by tracking the features derived from the image processing of chapter 2, whilst in the parameter setting experiment, the grid was placed by hand, and will be unaffected by any image processing errors.

If we now compare the tracking and parameter setting figures in table 7.4, we can see how these two experimental differences manifest themselves. For sequence A, the TPR is lower for the tracking experiment, and the FPR higher. Sequence A is arguably the most difficult image sequence for crop grid tracking, because many of the crop plant features are merged by the chain-coding algorithm, so the corresponding feature centroid does not represent a true crop plant position (see section 2.3.8). Poor tracking is almost certainly the reason for the drop in true positives and the increase in false positives.

In sequence B, the TPR is higher for the automatic tracker, and the FPR is the same as in the parameter setting experiment. The larger association region which is caused by the uncertainty in crop grid position is almost certainly responsible for the rise in TPR, as larger features from larger regions of the field are merged together. The FPR is unaffected in this case because of the very low weed density.

In sequences C and D, both TPR and FPR are higher for the automatic tracker than for the parameter setting experiment. In sequence C, the rise in both figures is most likely caused by the increased size of the association region. However, in sequence D, where strong shadows are present that will affect the quality of the tracking, the uncertainty on grid position combined with generally poorer position estimates will be the cause of the rise in both TPR and FPR.

The figures in table 7.4 show that the combination of size filtering and feature merging is very effective for classifying crop features, with true positive ratios in excess of 0.99 in for every sequence. The algorithm is less effective at weed pixel classification when tracking is difficult, as in sequences A and D, where the FPR rises to 15% and 20% respectively. This is not surprising because the success on the feature clustering algorithm hinges on the crop grid tracker providing good estimates of the crop position. However, when the tracking is easier, as in sequences B and C, the FPRs are much lower, 0.0% and 3.07% respectively.

7.3.2 Segmentation of real images

The second segmentation experiment relies wholly on the image processing algorithms of chapter 2 and tests the full segmentation algorithm illustrated in figure 7.3. In the experiment described in section 7.3.1 above, we knew that all the features presented for size filtering and clustering were true plant matter. In this experiment, some soil pixels will be misclassified as plant matter (and labelled C or W), and some plant matter pixels (crop or weed) will be labelled S.

Each of the tables 7.5 – 7.8 presents the percentage of the ground truth crop, weed and soil pixels classified as C, W and S, together with the total number of ground truth pixels in each class from the ground truth images of sequences A – D. The numbers of pixels bordering ground truth crop and weed features are also given as an indication of the number of doubt pixels that have been ignored in the classification totals. Each image is composed of 384×288 pixels, although only pixels in the region of the image that will pass underneath the autonomous vehicle are classified (see section 2.3.3).

		Classified as			Total number		
		C (%)	W (%)	S (%)	of pixels	Number of border pixels	
Ground truth	Crop	95.11	1.28	3.61	331,222	Crop	52,688
	Weed	10.50	51.88	37.62	505	Weed	1,373
	Soil	0.33	0.03	99.64	905,200		

Table 7.5: Run A segmentation results, percentages of true numbers of crop, weed and soil pixels classified as C, W or S, and the number of pixels that border crop and weed features. There are 16 ground truth images for sequence A.

		Classified as			Total Number		
		C (%)	W (%)	S (%)	of pixels	Number of border pixels	
Ground truth	Crop	78.90	3.72	17.38	53,514	Crop	19,615
	Weed	0.0	81.8	18.2	934	Weed	3,455
	Soil	0.01	0.04	99.95	1,152,254		

Table 7.6: Run B segmentation results, percentages of true numbers of crop, weed and soil pixels classified as C, W or S, and the number of pixels that border crop and weed features. There are 17 ground truth images for sequence B.

Perusal of the figures in tables 7.5 – 7.8 prompts a number of observations:

1. In every sequence, in excess of 98% of the soil pixels are correctly classified as S.
2. In each sequence, more crop pixels are misclassified as S than misclassified as W.
3. In each sequence, more weed pixels are misclassified as S than misclassified as C.
4. In sequences A and C, a greater percentage of crop pixels are correctly classified C than the percentage of weed pixels that are correctly classified as W.

		Classified as			Total number		
		C (%)	W (%)	S (%)	of pixels	Number of border pixels	
Ground truth	Crop	81.72	4.51	13.76	141,075	Crop	19,615
	Weed	3.93	56.90	39.17	17,160	Weed	18,544
	Soil	0.06	0.24	99.7	1,195,308		

Table 7.7: Run C segmentation results, percentages of true numbers of crop, weed and soil pixels classified as C, W or S, and the number of pixels that border crop and weed features. There are 17 ground truth images for sequence C.

		Classified as			Total number		
		C (%)	W (%)	S (%)	of pixels	Number of border pixels	
Ground truth	Crop	55.00	4.11	40.89	41,411	Crop	13,171
	Weed	6.31	73.52	20.17	1,046	Weed	2,202
	Soil	0.06	1.13	98.81	1,003,418		

Table 7.8: Run D segmentation results, percentages of true numbers of crop, weed and soil pixels classified as C, W or S, and the number of pixels that border crop and weed features.There are 16 ground truth images for sequence D.

5. In sequences B and D, a greater percentage of weed pixels are correctly classified W than the percentage of crop pixels that are classified C.

6. The number of doubt pixels that border ground truth weed features outnumber the total number of ground truth weed pixels in every test sequence.

7. The total number of ground truth crop pixels outnumber the doubt pixels that border the crop features in every test sequence.

Observations 1, 2 and 3 directly reflect the performance of the adaptive interpolated grey-level thresholding algorithm, which misclassifies a large percentage of the plant matter pixels as soil. This highlights the fact that the plant matter/soil discrimination problem requires more attention if image segmentation is to be improved.

Observations 4 and 5 suggest that the larger plants seen in image sequences A and C are more easily identified than the smaller plants in sequences B and D. The reasons for this are unclear, but may be related to changes in the infra-red reflectance of the crop plants as they age.

Observations 6 and 7 show that the weed features, which are dominated by border pixels, are typically smaller than the crop features. This has already been illustrated in figure 7.1 and forms the basis of the size threshold algorithm.

If we ignore the crop and weed ground truth pixels that the segmentation algorithm labels S, we can construct true positive and false positive ratios for the crop and weed pixels that have been classified as plant matter (either C or W). These figures are given for each sequence in table 7.9 and show that those pixels which *are* identified as plant matter are separated into the crop and weed classes with some success. This allows us to conjecture that if plant matter/soil discrimination were more reliable then figures similar to those in table 7.4 might be obtained.

Sequence	TPR	FPR
A	0.9639	0.1683
B	0.9550	0.0
C	0.9477	0.0650
D	0.9305	0.0790

Table 7.9: TPR and FPR for the correctly identified plant matter pixels in sequences A–D.

7.4 Summary

We have presented a three stage algorithm (figure 7.3) for segmenting images captured by the autonomous vehicle into three classes, soil (S), weed (W) and crop (C). The first stage of the

algorithm is the adaptive interpolating grey-level threshold followed by pixel clustering which separates the image into soil pixels and plant matter features, and was described in section 2.3. The second stage exploits the fact that the crop plants are typically larger than the weeds, and filters the plant matter features on the basis of size. Those falling below a size threshold are classified as weed, those above the threshold are passed on to a clustering algorithm which groups together features which lie close to the crop plant positions predicted by the vehicle's current state estimate.

Each stage of the algorithm is controlled by a parameter. The adaptive interpolating threshold algorithm is controlled by a gain parameter, α, and this was set for each test image sequence in chapter 2. The feature size threshold is determined by a parameter ς, and the clustering algorithm by proximity radius r. To select values for ς and r, the maximum realisable receiver operating characteristic (MRROC) curve may be used. In the absence of information concerning the specific costs and values of the classification system, operating points which maximise the true positive ratio and minimise the false positive ratio were chosen for each of the four test sequences.

Two experiments have been performed to test the segmentation algorithm. In both experiments, the crop grid tracking algorithm[3] was run fully automatically on each sequence. In the first experiment, the size filtering and feature clustering algorithms were used to differentiate between the ground truth weed and crop features for each sequence. The rates of correct crop classification and weed misclassification were compared with those expected from the operating point of the algorithm that was set manually, assuming (optimistically) perfect knowledge of the crop grid position. The segmentation algorithm was found to be more effective at correctly identifying crop pixels than rejecting weed pixels, especially on the more challenging image sequences with shadows or large crop features that merge together in the image.

The second experiment tested the size filtering and feature clustering algorithms on the features derived by the image processing techniques of chapter 2, resulting in images where the pixels were labelled as one of three categories S (classified as soil), W (classified as weed) and C (classified as crop). These labelled images were compared with the ground truth soil, weed and crop segmentations. From this test it was concluded that the majority of the classification errors were plant matter pixels being labelled S. Of those plant matter pixels correctly classified as plant matter, the discrimination between crop and weed pixels reflected that found in the first experiment, that was performed on ground truth features.

The segmentation algorithm presented in this chapter would appear to be effective at cor-

[3] AUTO2 of chapter 5.

rectly identifying crop pixels, although the number of misclassified weed pixels is of concern. Although the image processing techniques presented in chapter 2 have been able to provide features to support crop grid tracking, the experiments performed in this chapter have shown that they are not so effective for the image segmentation task, and improved discrimination between plant matter and soil would be desirable.

The next chapter demonstrates the on-line system at work in the field. Navigation trials similar to those conducted in the test-bed environment in chapter 5 are carried out in a field of crop, and a demonstration of crop treatment is given.

Chapter 8

Field trials

The success of the computer vision algorithms has been demonstrated tracking both off-line image sequences and in a simplified indoor test-bed environment (chapter 5), and also in crop/weed discrimination on sequences analysed off-line (chapter 7). The next step is to transfer the technology into the field environment to be tested "live" on a bed of crop. This will provide an opportunity to gain further data on the system performance in terms of navigation and treatment accuracy. The on-line implementation of the vision system was converted from the off-line system that was demonstrated in the first part of chapter 5 by Tony Hague, who also designed and built the dead-reckoning estimation system and vehicle controller [HT96].

The following sections describe two experiments. The first experiment is designed to analyse the performance of the on-line navigation system, and is similar to the trials performed on the indoor-test bed, but on a field of crop. The second experiment focuses on crop treatment, where the vehicle sprays crop plants with a blue marker dye, thereby allowing analysis of the accuracy of treatment and also the selectivity of the treatment algorithm, i.e. the number of crop plants missed or weeds incorrectly targeted. We should recall from chapter 1 that treatment application is the central aim of plant scale husbandry, so the results from these trials will show how close we are to achieving this goal.

8.1 Position estimation

Chapter 5 described experiments designed to test the accuracy of the vehicle's position estimates, the results of which demonstrated the effectiveness of the system on both image data captured from the vehicle, and in a simplified test-bed environment where the image processing task was undemanding. The results showed that the crop grid tracking algorithm of chapter 4 is suitable for the vehicle localisation task. It would, however, be inappropriate to simply assume that the algorithm will perform equally well as part of a closed-loop vehicle controller in the more demanding field environment, so the test experiments must be repeated in the field.

The experimental method used is similar to that given in chapter 5. A trail is left on the ground by the vehicle, and its offset relative to the crop rows is compared to the estimated position logged by the navigation system. The tracking algorithm implemented in the on-line system does not estimate the crop grid parameters $\bar{r}$ and $\bar{l}$, and we have already seen in chapter 5 that this algorithm will produce biased estimates of forward position if the true grid parameters diverge from those used by the algorithm. The dead-reckoning system is also subject to increased error in the field as wheel-slip is more likely on soil than indoors. Wheel-slip causes inaccuracies in the information delivered by the odometers, and these errors will filter through to the position estimate.

Moving from the simple indoor test-bed to the outdoor field introduces problems for the image processing and computer vision algorithms as well as for the dead-reckoning system. The test-bed environment is comprised of a simple scene containing white circles arranged in a regular grid on the black ground plane to mimic the crop planted in the field. The field environment is more complex for a number of reasons. Firstly, the crop plants are no longer flat objects lying on the ground plane but three-dimensional, so the projection errors generated when calculating the position of image features on the ground plane will occur, as noted in section 2.3.8. Secondly, the presence of weeds adds clutter to the observation data, so the validation technique of chapter 6 is required. Thirdly, the contrast between plant matter and the soil is not as strong as that between the white circles and black mat of the test-bed. In addition, lighting artefacts such as shadows in strong sunlight may be present, causing single plants to fracture into multiple image features. Finally, the grid pattern formed by the crop plants is less regular than that of the pattern painted on the mat, so the fixed parameter crop grid tracker is almost certain to produce biased estimates.

A further difficulty with field operation involves performance assessment. When we attempt to assess the system's performance, problems arise from the inherent uncertainty of the position on the ground plane of the grid of crop plants, and hence the position of the trail left by the vehicle relative to this grid. In the indoor test environment, the circles representing the crop were positioned in a regular grid, and relative to a known origin. Outdoors, the crop are less regularly planted and grow asymmetrically, which leads to the grid structure becoming more approximate, and measurement of the "centre" of the plants more difficult. The rows in which the crop lies are not necessarily straight, although they may be assumed to be so locally. In addition, there is certainly no clear, fixed origin with respect to which the grid of crop plants may be localised, and hence no clear origin relative to which the trail left by the vehicle should be measured. To counter the uncertainty in the position of the central row of plants in the crop grid,

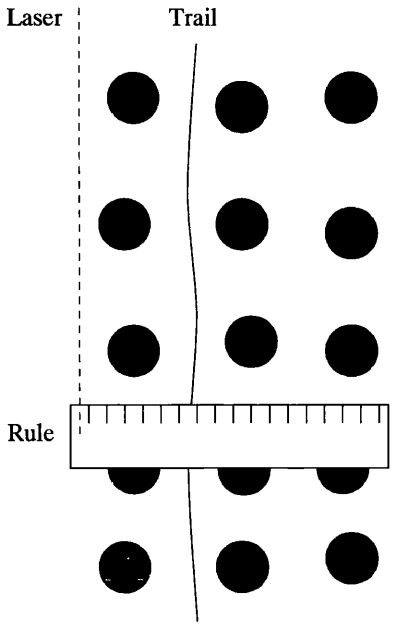


Figure 8.1: Measuring the plant and trail positions.

the position of the trail is measured with respect to the mean position of the crop plants in each line of the crop. A laser beam, shone approximately parallel to the central row of plants, as illustrated in figure 8.1, is used to provide an origin which aids consistent measurement of the plant and trail positions. At each line of plants (approximately every $0.5m$ along the crop row), a rule (with retro-reflective target attached to increase laser visibility) was aligned with the laser beam, and the position of the root of each plant read off. The position of the trail on the rule was also noted. The thickness of the plant roots, together with the width of the dye trail left by the vehicle, means that these measurements are of limited precision (*c.* 5 millimetres), which will add noise to any performance metric derived from them. The experimental arrangement is illustrated in figure 8.1, where the crop are shown as black circles, the trail left by the vehicle as a solid black line, and the laser beam a dotted line.

As in chapter 5, the performance is calculated in terms of the root mean square error between the measured points on the dye trail and the corresponding estimated position held by the vehicle controller. This metric was used by Hague *et al* [HMT97a] to measure the system performance when the Hough transform vision algorithm of Marchant and Brivot [MB95] was used. A single run with this system over $25m$ of crop was found to have a r.m.s.e. figure of $8mm$. A similar experiment conducted by Southall *et al* [SHMB99], also using the Hough Transform algorithm, yielded a r.m.s.e. figure of $13.5mm$. The reasons for this difference are unclear, although factors such as experimental errors and crop condition will play their part. Whatever

the reasons, it should be added that a localisation accuracy of $13.5mm$ is certainly sufficient to avoid accidentally driving over the crop. In addition to the r.m.s. error, we will also calculate the mean error to check for bias, as we did in the test-bed experiment.

8.1.1 Experimental results

Three beds of crop, each consisting of three rows of six week old cauliflower plants were used to test the crop grid tracking algorithm, and the vehicle was programmed to traverse each $40m$ long crop bed twice (once in a North-South direction, once South-North), resulting in six sets of measurements. Figure 8.2, parts (a)-(f) plot these, where the solid line marks the estimated offset of the vehicle from the central row of crop, and the dashed line shows the measured trail offset. It should be noted that because the measurements are taken only at every line of plants in the bed, there may be discrepancies between the total length of the measured and estimated traces in each plot. Unfortunately, owing to an experimental oversight, the position variances for these runs were not logged, although data from similar outdoor trials where the variance was logged showed that the filter settles down to a steady-state standard deviation of approximately $2.25mm$. This value is lower than the typical error estimates from the indoor trials (c. $3.1mm$), a factor that may be attributable to observation noise and the use of the "virtual ground plane" to reduce perspective projection errors. In chapter 4, section 4.2.5, we showed how the observation noise covariance matrix $\mathbf{R}$ associated with an observed feature location in the image is derived by projecting a covariance matrix $\mathbf{R}_w$ from the ground plane into the image plane. To reduce the effects of projection errors on object centroid location in the field, we use a virtual ground plane as discussed in section 2.3.8, set at the height of the crop plants[1]. In this case, the covariance matrix $\mathbf{R}_w$ is projected from the virtual ground plane into the image which, because the virtual ground plane is nearer to the camera than the true ground plane used as a reference in the test-bed experiments, leads to a smaller observation covariance $\mathbf{R}$ than in the test-bed trials. This in turn will lead to a larger Kalman gain (because $\mathbf{R}$ is in the denominator of equation 3.14), and hence a reduced covariance estimate.

For the six runs illustrated in figure 8.2, the root mean square errors on the offset estimate are given in table 8.1. Over all 6 runs, the r.m.s.e. between each measurement/estimate pair in all the data sets is $15.3mm$, a level of accuracy that is easily sufficient for safe navigation along the crop rows. Whilst this mean r.m.s.e. is slightly greater than those given for the previous Hough transform algorithm ($13.5mm$, $8mm$), it should be noted that the crop grid tracking algorithm provides more information (forward distance estimates and crop/weed discrimination) than the Hough transform row tracker. A small degradation in tracking performance is thus a

[1] Which we estimate by eye.

reasonable trade-off for the extra information. As would be expected, given the increased difficulty in navigation and measurement in the field, the r.m.s.e. figures in table 8.1 are higher than those obtained from the indoor tests given in chapter 5.

The algorithm not only shows satisfactory mean square performance, but the mean difference between estimated and measured position over every measurement/estimate pair in all six data sets is $0.55mm$, so the bias of the lateral offset estimate is negligible. The estimates of

Track	r.m.s.e. (mm)
(a)	16.7
(b)	16.5
(c)	13.4
(d)	16.9
(e)	13.2
(f)	14.5

Table 8.1: The root mean square error between the measured and estimated offsets for the six experimental runs of figure 8.2.

forward distance are given in table 8.2 and, as we might expect, are much less accurate in the outdoor environment than on the indoor test-bed. The largest error on the test-bed, using the crop grid tracker, was $9cm$ on a total distance of $12m$ ($0.75cm$ error per metre travelled), whilst in the field environment the largest error is $-1.7m$ on a total distance of $41.6m$ ($-4.1cm$ error per metre travelled). The degradation in performance can again be attributed to the increased difficulty of image processing and dead-reckoning in the field. We should also remember that the crop grid tracking algorithm implemented on-line does not estimate the grid spacing parameters $\bar{r}$ and $\bar{l}$. This algorithm was seen to produce inaccurate estimates of forward distance travelled in chapter 5, so the bias noted in table 8.2 is not unexpected. The fact that the estimated distance travelled is always under-estimated suggests that the grid parameter $\bar{l}$ in the algorithm is too small, although this has not been verified by measurement in the field. We believe that estimating $\bar{r}$ and $\bar{l}$ will reduce these errors.

8.2 Crop treatment

To demonstrate the utility of the crop grid tracking algorithm for crop treatment, the vehicle was programmed to navigate a 12 metre section of 9 week old crop at a speed of 0.6 ms^{-1}, spraying a blue dye onto the crop plants, whilst leaving weeds untreated. This experiment tests both the segmentation performance of the algorithm and the accuracy of the position estimation,

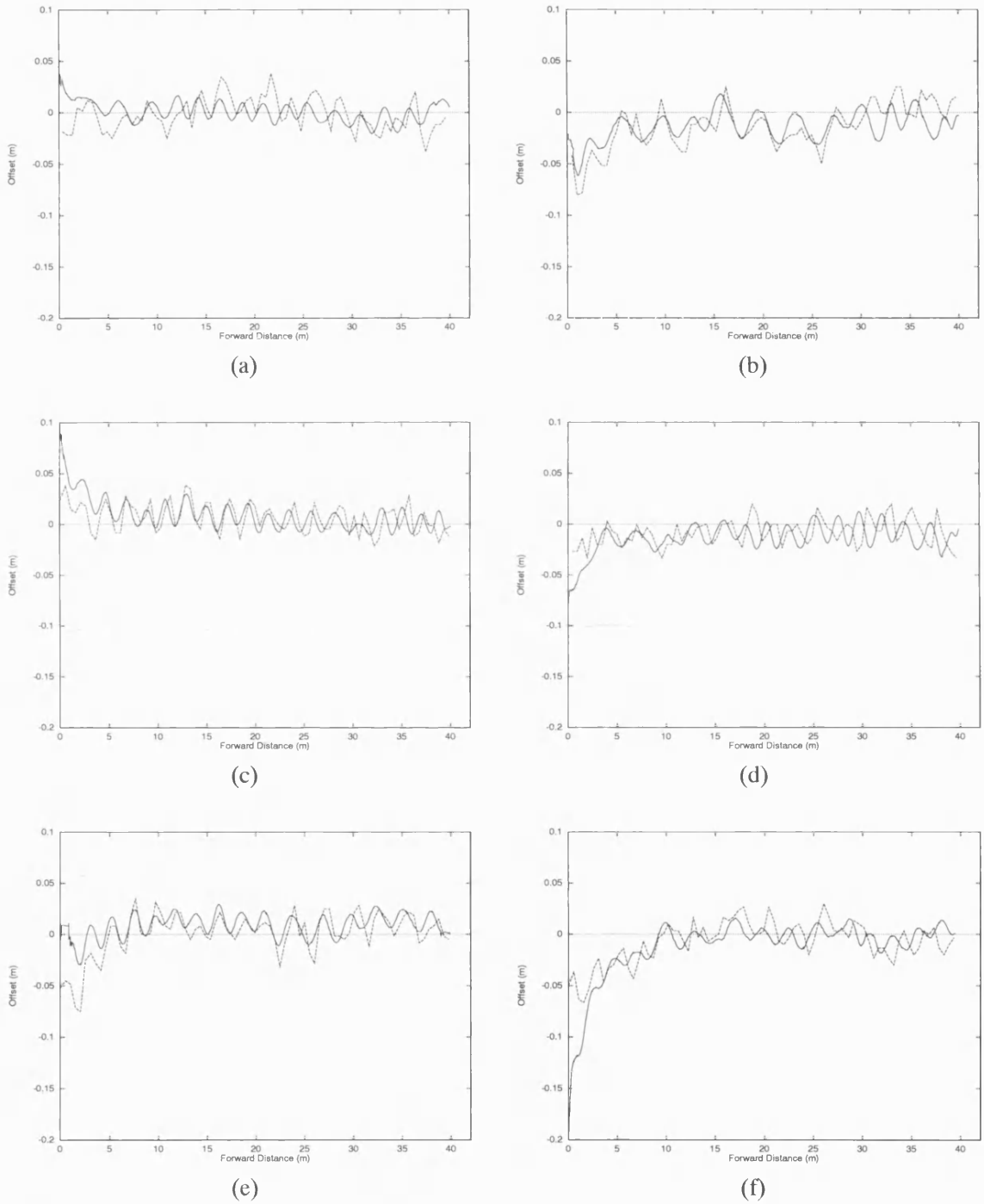


Figure 8.2: Navigation trials. Each figure (a)-(f) shows the estimated (solid) and measured (dashed) offset of the vehicle from the central row of crop.

Track	Measured (m)	Estimated (m)	Error (m)
(a)	41.3	40.02	-1.28
(b)	41.5	40.06	-1.44
(c)	41.3	40.05	-1.25
(d)	41.2	39.93	-1.27
(e)	41.6	39.90	-1.7
(f)	41.05	39.45	-1.6

Table 8.2: Measured and estimated forward distance.

and mimics the operation of a plant scale husbandry system. Segmentation performance can be assessed by simply counting the number of crop plants missed and weeds incorrectly treated, whilst positional accuracy is reflected by the relative position of the plant centres with respect to the regions of earth sprayed.

Before reporting the experimental results, an outline of the on-line segmentation system should be given. The feature classification algorithm described in chapter 7 has not been fully implemented in the on-line system. There is also a mapping algorithm, designed and implemented by Hague [Hag99], which places classified image features into a map of the ground plane. The map activates the treatment system as the crop features pass below the vehicle's spray bar. Furthermore, a map of the field showing crop and weed growth is a useful tool in precision agriculture, allowing assessment of both crop and weed growth.

Like the off-line classifier, the on-line feature classification algorithm is a two stage process where image blobs are first filtered on the basis of size before being processed by a data association step. Unlike the off-line algorithm, however, the data association is performed using the predicted crop plant positions (c.f. chapter 6), rather than the updated plant positions (c.f. chapter 7). The feature merging process described in chapter 7 was also not implemented.

The mapping algorithm constructs a map of weed and crop coverage of the ground plane in front of the vehicle over a field of view approximately 2.5 m ahead. The map is used to control the treatment system, which consists of a row of spray nozzles whose location relative to the ground plane projection of the image centre[2] is known. As the vehicle traverses the field, the nozzles pass over the map. When a nozzle is over part of the map scheduled for treatment, it is activated. Conceptually, the map is constructed by using the camera calibration and vehicle position estimate to project features from the image onto the ground plane although in practice, for

[2]This is where the camera's optic axis intersects the ground plane, as illustrated in figure 2.4 in chapter 2.

computational speed, the bounding box of each feature is projected from the image to a quadrilateral area of the map (this will not be square, owing to the perspective projection between image and map). The map itself consists of a 2D array of counters 256 elements wide, 3000 elements deep, and each element in the array representing a patch of ground $13.33mm$ square, for a total map length of 40 metres, and width of 3.4 metres. When a feature's bounding box is projected on to the map, the counters in the map elements lying in the projected area are incremented. If the feature has been classified as crop, then the "crop" counter is incremented, otherwise the "weed" counter is incremented. At any time, the classification of a map element is given by the largest counter total, if that total exceeds a threshold value. If both counters are below the threshold level, the element is taken to represent soil, because insufficient plant matter has been detected there.

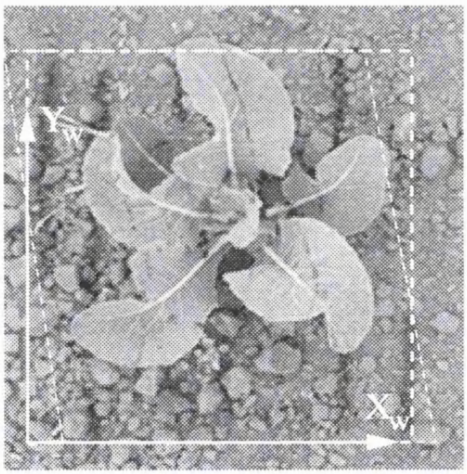


Figure 8.3: Spray measurement. The axes mark the directions of x_w and y_w, the world coordinate axes. The thicker dashed line shows the bounding box of the sprayed area on the ground, whilst the thin dashed line shows the rhomboid formed by projecting the corresponding feature's bounding box from the image onto the ground plane.

To assess the accuracy of crop spraying and the success rate in correctly identifying crop plants, the vehicle was programmed to navigate a 12 metre plot of crop, and to only spray regions of the map marked as crop plants. The spray system was loaded with a blue dye solution which would leave traces on the plants and ground to enable the accuracy of the spray application to to be measured.

Figure 8.3 shows a crop plant that has been sprayed with dye, which appears as dark lines on the soil and across some of the plant's leaves. Two dashed outlines and a set of axes have been

superimposed on the photograph. The lighter dashed rhomboid shows the shape of the bounding box projected from the image onto the map of the ground plane. The heavier dashed square is the bounding box of the region of earth that has been treated. The axes mark the directions x_w, y_w, where y_w is parallel to the central row of crop. We determine the accuracy of spray localisation by measuring the distance, in both the x_w and y_w directions, between the centre of the heavier dashed box and the plant's root. These reference points are taken for ease of measurement, but it should be borne in mind that the position of the plant root does not necessarily reflect the centre of the plant's "footprint" on the ground plane, nor does the centre of the heavier dashed box necessarily represent the vehicle's estimate of the plant centre. The resulting measurements do, however, give us an idea of how well the vehicle localises individual plants on the ground plane, which is a different problem to localisation with respect to the group of plants in the crop grid, as tested in the navigation experiments above.

8.2.1 Experimental results

Figures 8.4 and 8.5 show the two halves of a photographic montage taken after the experimental treatment of a twelve metre patch of crop together with the corresponding map generated by the vehicle. The map, which has been resized by eye to match the scale of the photograph[3], shows soil regions as black, weed as red and crop as green. As would be expected, most of the features in the map are rhombic in shape owing to the perspective projection of image feature bounding boxes onto the ground plane. The bounding box approximation leads to some excess spray being applied to the soil, so for precise treatment of plant matter alone, a more faithful representation of the plant outline must be used. However, we can see that much of the bare soil is left untreated.

As we can see from the photographs, all of the crop features have been treated, and none of the weeds. Perusal of the map allows us to form an overall impression of weed recognition performance, whilst figures 8.6 and 8.7 highlight some typical successes and failures of weed identification. In the first example (figure 8.6), we can see a correctly classified weed (top left of the image), and also a stone that has been incorrectly classified as weed matter. The stone has been misclassified because it is a bright object in the infra-red image. More sophisticated image processing using colour in addition to near infra-red information could eliminate such errors. It is also worth noting that in the map shown on the right of figure 8.6, the red weed feature that represents the stone is touching the crop plant, whilst the two are quite separate in the photograph. This is likely to be an artefact of using the virtual ground plane as proposed in chapter 2. Whilst using the virtual ground plane reduces the projection errors for crop plants that stand

[3] Unfortunately, the camera used to take the photographs shown on the left of figures 8.4 and 8.5 was not calibrated, so accurate matching of photograph and map scales is not straightforward.

proud of the true ground plane, it will *increase* projection errors for objects, such as the stone or small weeds, that lie close to or on the true ground plane. The second example (figure 8.7) shows part of a crop plant that has been incorrectly classified as weed, an error symptomatic of the fracturing of plants into multiple leaf objects during the image processing. It may be possible to reduce such errors by using the feature clustering algorithm of chapter 7, but this has yet to be implemented on-line.

Quantitative measures of the accuracy of the treatment system were derived by comparing the centres of the sprayed regions on the field with the root positions of the corresponding plants, as described above (figure 8.3), to derive the positional error in both the x_w and y_w directions. The mean error and root mean square error in each direction are given in table 8.3. Perusal of the mean error figures shows that the plant position estimates are biased in both x_w and y_w, although it should be noted that the bias in the x_w direction, $9.7mm$ is smaller than the resolution of a map element (13.33 mm). We can relate the bias on y_w measurement to the figures shown in table 8.2. We have 24 plants in the twelve metre bed, so the total error over 12 metres is $-55.44cm$, which scales up to an error $-1.85m$ over $40m$, which is a just little larger than the worst errors seen in table 8.2.

Direction	Mean Error (mm)	RMS Error (mm)
x_w	9.7	33.6
y_w	-23.1	34.4

Table 8.3: Spray accuracy results.

The root mean square errors in both directions are in the region of $34mm$, which shows that there is a significant (compared to the mean) variation in the error measure. This variation arises from a number of sources. The first source is the measurement process. As noted above, the plant root is not necessarily the true centre of the plant's footprint on the soil. A second source of variation is related to the fact that the grid parameters are fixed throughout tracking. In chapter 5 we saw that when this tracker's fixed parameters were in agreement with the true values, the algorithm performed well with small bias, and when the grid parameters differed from those values fixed in the algorithm, that biases were introduced. Thus, as the crop plant grid spacings vary across the field, tracker bias errors will be introduced, increasing the r.m.s. error.

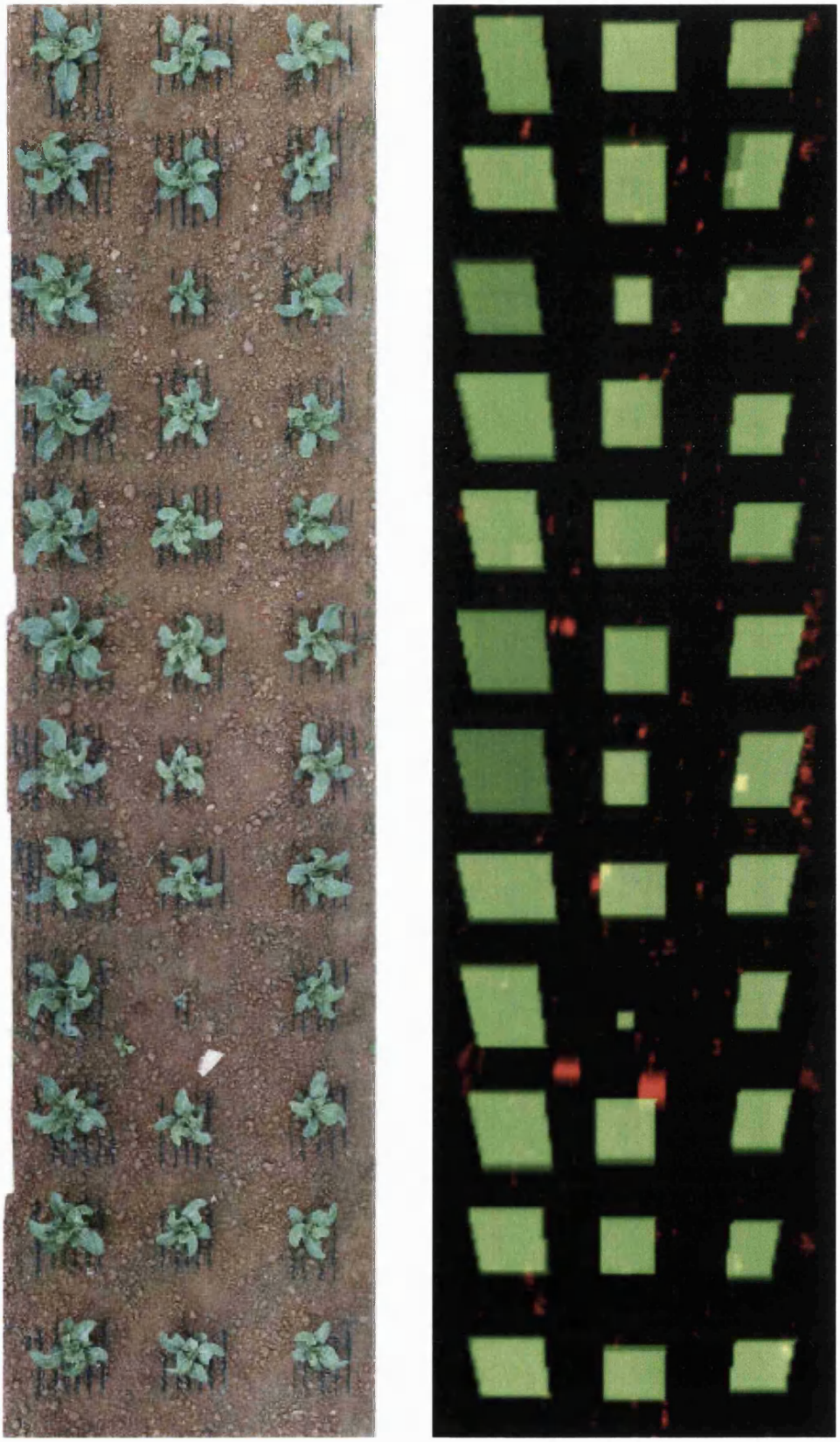


Figure 8.4: Spray treatment results, part one. Left: treated field. Right: Map – black = soil, red = weed, green = crop.

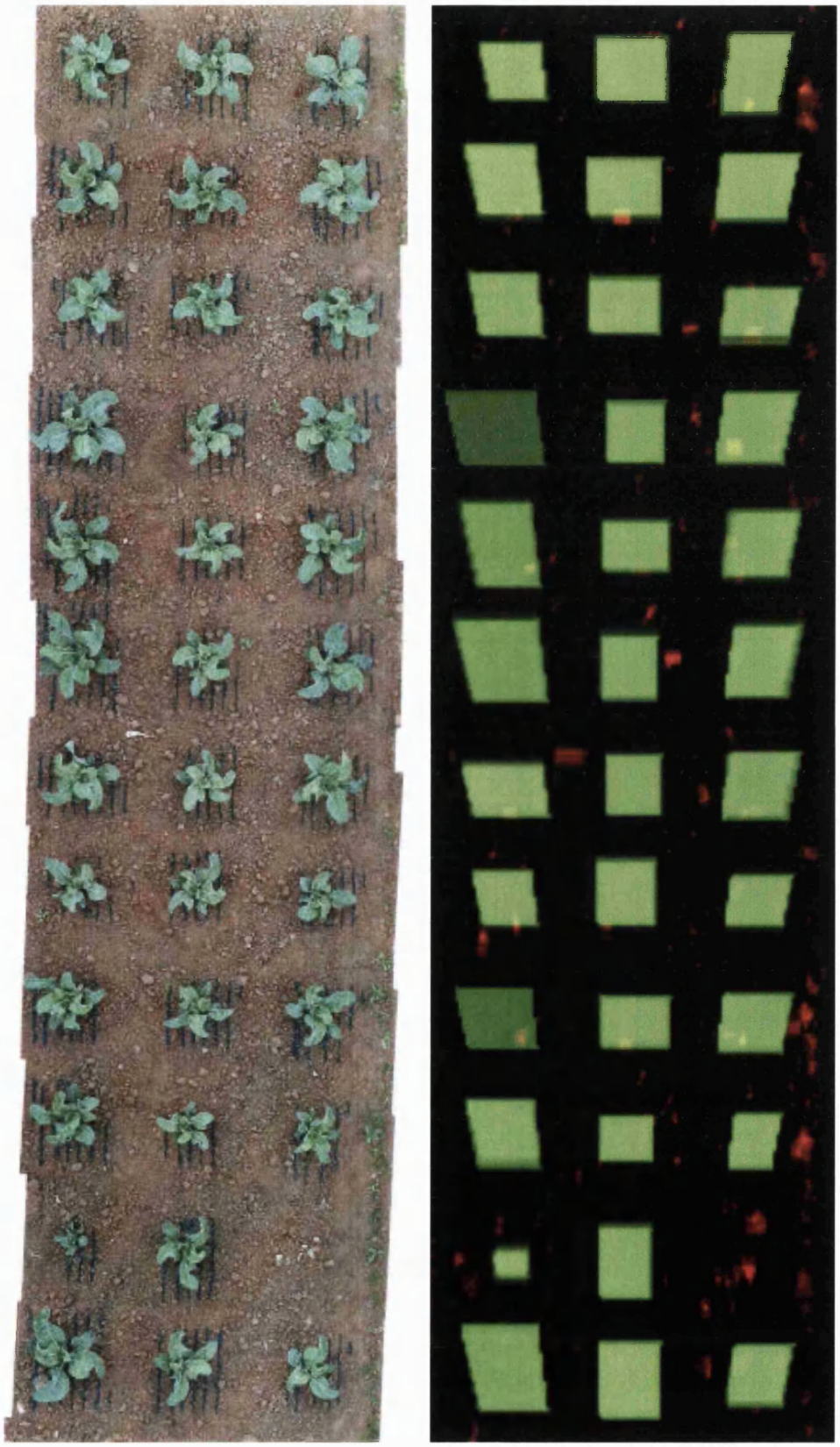


Figure 8.5: Spray treatment results, part two. Left: treated field. Right: Map – black = soil, red = weed, green = crop.

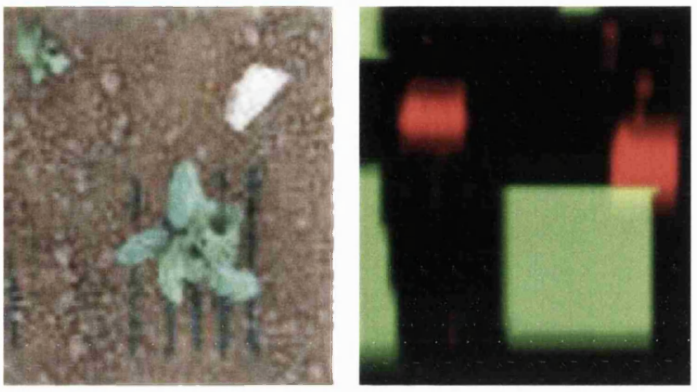


Figure 8.6: Weed identification, example one. In this image, a correctly classified weed (top left) is seen, together with a stone that has been classified as a weed.

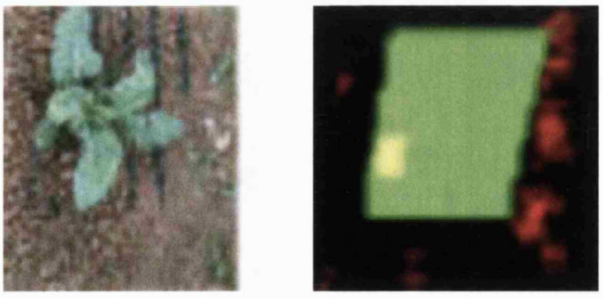


Figure 8.7: Weed identification, example two. This image shows a crop plant bounding box which contains a region that has been incorrectly classified as weed. This region is, in fact, a leaf of the plant that has fractured from the plant body as a result of the image processing.

8.3 Summary

Trials have been performed in the field to assess both vehicle navigation and crop treatment using the fixed parameter crop grid tracker. The navigation trials performed outdoors yield root mean square errors in the region of $15mm$ on the offset from the central row, which is perfectly acceptable for navigation purposes. Motion in the forward direction is underestimated, typically by about $3cm$ error per metre travelled. In chapter 5, we saw that an error of approximately $8.2cm$ per metre could accumulate. The lower figure of $3cm$ per metre seen here may be explained by the fluctuating crop grid size (the tracker will perform better in regions of crop where the grid size is closer to its own model), or perhaps even wheel-slip, which will lead to over-estimation of the distance travelled by the dead-reckoning system, and may be compensating for underestimates from the vision system. This requires further investigation, including implementation of the tracker that estimates the crop grid parameters (algorithm AUTO2 from chapter 5).

The bias of the forward distance estimate is also seen in the crop treatment experiment where there is a bias on the estimated longitudinal position of the crop plant centres, with a mean value of $-23.1mm$. The negative value indicates an underestimate. There is also a small bias in the lateral position of the crop plants, although at $9.7mm$, this is smaller than the resolution of the map. For the purposes of crop plant treatment, errors such as these should be acceptable, as they represent only a fraction of the size of the crop plant. For the treatment of weeds, which are often much smaller than the crop plants, higher precision is likely to be required.

Qualitatively, the treatment system appears to work well. All of the crop plants were correctly sprayed, and none of the weeds. Perusal of the map produced by the vehicle, and comparison with the photographic montage of the treated field, shows in addition that most of the weeds were correctly identified, although there are, as expected some examples of crop leaves being classified as weed, and some non-plant matter objects (stones, soil highlights) incorrectly classified as weed. Future work needs to address the issue of weed treatment and how to improve the resolution of the map and the precision of the description of the plant outlines in the map to reduce over-spraying.

Chapter 9

Conclusions

In chapter 1, we introduced the concept of plant scale husbandry (PSH), and motivated its use for economic and ecological reasons. The need for autonomous systems to apply this level of treatment was underlined, and we presented the Silsoe autonomous horticultural vehicle as a test-bed for experiments in PSH. It was stated that the aim of this thesis was to extend the capability of the autonomous horticultural vehicle by introducing a vision algorithm centred upon the use of a crop grid model, which would allow two principal advances. We repeat these here:

1. Forward distance estimates may be obtained by a vision algorithm that tracks the crop grid model. In the original autonomous vehicle navigation system, estimates of forward distance are generated by dead-reckoning alone, and the row model used in the Hough transform vision algorithm does not permit estimation of forward motion. Dead-reckoning systems are prone to accumulating errors that lead to biased position estimates, and problems with wheel-slip can lead to unreliable odometer measurements in the field. It is hoped that the addition of vision estimates of forward distance will correct for dead-reckoning bias.

2. The crop grid model may be used as an aid for discriminating between crop and weed plants. For reasons we shall outline in chapter 2, discriminating between crop and weed plants with image processing techniques is a much harder problem than extracting both weed and plant features from the images. However, if we are successfully tracking the crop grid, then image features that support the estimate of grid position may be assumed to be crop plants, and the remainder weeds.

The following two sections will address these points, and we will state the contributions made by this thesis toward localisation and plant treatment, and review the success of the algorithms developed. Open questions and directions for further work are suggested.

9.1 Localisation

We have developed a computer vision system that uses an extended Kalman filter to track the grid pattern formed by the crop in the field.

- Two alternative crop grid tracking algorithms have been implemented and tested off-line (chapters 4 and 5). In chapter 4, two different models of the crop grid were proposed, one where the grid parameters $\bar{r}$ and $\bar{l}$ were fixed, and another where they were estimated by the Kalman filter. The algorithms based upon these two models were tested off-line and their position estimates compared to human assessments and the output of Marchant and Brivot's Hough transform method. It was found useful to estimate the grid parameters on-line to allow for local variations in the planting pattern and to reduce the bias in forward distance estimation. We saw in chapter 5 that the tracker with fixed grid parameters accumulated an error in the region of $76mm$ over a distance of $c.1m$ when compared with ground truth data produced by hand. The algorithm that does estimate the grid parameters was within $9mm$ of the human assessment, a considerable reduction in bias.

- The algorithm with fixed grid parameters was tested as part of the on-line system, first of all in a simplified test-bed environment (chapter 5) and secondly in a field of crop (chapter 8). The test-bed experiment showed that lateral position estimation was accurate to within an r.m.s. error of $8mm$ and forward distance estimation to within $0.75cm$ per metre travelled (over a $12m$ run). The field experiments, where accurate measurement is much more difficult than in the test-bed, yielded larger errors, with r.m.s. error of $16mm$ on lateral position, and forward distance underestimates of up to $-4.1cm$ per metre travelled (over a $40m$ run). The precision of the lateral estimation is easily adequate for safe navigation along a row of crop, and satisfies the $20mm$ accuracy estimated to be required for treatment operations by McLellan and Friesen [MF96]. The reduced bias figures quoted above lead us to believe that estimating the grid parameters will reduce forward distance estimate error.

- The control theoretic issues of controllability and observability and their implications for Kalman filtering have been discussed (chapters 3 and 4). A new test for the observability of the linearised system in the extended Kalman filter has been derived and used to confirm the observability of the extended Kalman filters used in the crop grid tracking system. The concept of "corruptibility" has been introduced to describe the transfer of noise inputs into a system's state variables, and it has been demonstrated that a process model that is partially incorruptible is desirable for estimating the grid parameters $\bar{r}$ and

$\bar{l}$. Partial incorruptibility refers to the fact that process noise is added to the estimates of $\bar{r}$ and $\bar{l}$ only when new information about these parameters becomes available, and for the majority of the time they are incorruptible. The usual Kalman filtering process assumes full corruptibility, with process noise added at each iteration (sections 3.4.5 and 4.4).

- An algorithm for initialising the extended Kalman filter has been proposed that includes a novel use of the discrete Fourier transform for estimating the position of the crop plants within a row (chapter 6). Although the algorithm does not automatically generate an initial covariance matrix for the extended Kalman filter, this has been estimated by analysing the algorithm's performance on test sequences.

9.2 Plant treatment

The concept of plant scale husbandry is centred upon the differential treatment of individual crop and weed plants, and the use of the crop grid model in this thesis has made significant steps toward this aim.

- An algorithm has been proposed to discriminate between crop and weed plants on the basis of their size and position in the field (chapter 7). The algorithm has been tested off-line on four image sequences captured from the vehicle that contain different stages of crop growth and differing imaging conditions. When tested on plant matter features extracted from the images by hand, the algorithm is found to provide good separation of crop and weed features, although the performance degrades when the plant matter features are provided by the adaptive interpolating threshold algorithm described in chapter 2. The segmentation results were inferior in the sequences that contained shadows and plants that touch each other in the image. The former makes detection of plant matter difficult whilst the latter violates the assumption that each image feature corresponds to a single plant.

- A similar algorithm has been implemented in the on-line system and used in a treatment experiment in the field that emulates plant scale treatment of real crop (chapter 8). The vehicle was programmed to spray a blue dye on regions of the field it had classified as crop plant, and to leave weeds untreated. Measurements were taken to compare the centroid positions of the sprayed regions with the root positions of their corresponding plants, and an r.m.s. error of under $35mm$ was found in each direction along and perpendicular to the rows of crop. This is sufficient to lead to significant savings in crop treatment and reduction of adverse environmental contamination. However, it is likely that greater accuracy is required for precise treatment of weeds. The field trials demonstrated that no

weeds were incorrectly treated, and measurements taken from the photographs in figures 8.4 and 8.5 indicate that only one third of the region between the vehicle's wheels was treated with dye. In more traditional spraying, the entire field would be treated, so there is a saving of applicant of 67%. The current system uses a crude approximation of the plant boundary to determine the region of field to be treated, and we believe that a more accurate description would reduce applicant quantities even further.

9.3 Further work

Whilst we believe that the results presented in this thesis have demonstrated the usefulness of the crop grid structure in plant scale husbandry, there are a number of open issues that require further investigation, some of which are very specific to the PSH task, others of a more general nature. We have restricted the suggestions below to development of the computer vision system, but it should be noted that there is scope for much research in dead-reckoning and vehicle control, and in the design of spray systems for plant scale treatment.

We have shown on off-line sequences that the tracking algorithm which estimates the grid parameters $\bar{r}$ and $\bar{l}$ provides better estimates of longitudinal position. This algorithm has yet to be implemented on the autonomous vehicle, but we believe this will be necessary to reduce the forward distance estimation errors that were measured in the field trials, and that estimation of the grid parameters is necessary to take account of the variation of the planting pattern in the field. Even with the improved tracking algorithm, it is unlikely that localisation accuracy will be sufficient for precise application of chemical spray to small weeds. An alternative to spraying is the use of mechanical weed control, for example an automatically guided hoe, which has additional environmental benefits and is likely to require less precise position estimation than spray application.

The clustering algorithm described in chapter 7 has also yet to be implemented in the on-line system. However, a general point about the crop/weed segmentation algorithm is that its performance is controlled by three parameters (a grey-level threshold gain, a size threshold and an association radius) which are currently set by hand on the basis of ROC analysis. In a commercial system, it would be desirable to control these parameters automatically, so that the farmer need not understand the underlying algorithm in order to use the vehicle system. Automatic selection of algorithm parameters in complex systems such as ours is an open issue in computer vision [PDK96, Mar98], and is likely to become more important as such systems leave the laboratory and enter the marketplace.

In the off-line segmentation experiments of chapter 7 it was seen that the grey-level thresh-

olding technique used to extract plant matter from the images leads to a large number of crop and weed pixels being misclassified as soil. Andersen, Marchant and Onyango have recently proposed a "dedicated sensor" for reliably detecting plant matter in outdoor scenes [AOM99, AMO99]. The sensor is similar to a standard RGB colour camera, but the blue channel is replaced by near infra-red. Use of such a sensor may lead to more reliable discrimination between plant matter and soil, provided that algorithms can be devised which cater for the effects of changes in daylight illumination on images formed by the sensor. Finlayson *et al* [FSC98] offer an algorithm for colour normalisation to remove effects of illuminant colour and geometry in RGB images, where they assume that illumination is constant across the scene. This assumption does not apply in outdoor scenes, where the illuminant is a mixture of two sources, skylight and sunlight [AMO99], so a different approach will be necessary.

Further effort could be directed toward designing image processing algorithms that could discriminate between crop and weed plants, and indeed identify separate crop plants once they have grown large enough to touch each other in the image. As we noted briefly in chapter 2, the variability between plants of the same species makes modelling the crop and weed plants very challenging. Any techniques that would be of use in this problem would almost certainly find widespread application in many problem domains.

Bibliography

[AF88] N Ayache and O D Faugeras. Building, registering and fusing noisy visual maps. *International Journal of Robotics Research*, 7(6), 1988.

[AF89] N Ayache and O D Faugeras. Maintaining representations of the environment of a mobile robot. *IEEE Transactions on Robotics and Automation*, 5(6):804–819, December 1989.

[Ale98] D Alexander. *Statistical Modelling of Colour Data and Model Selection for Region Tracking*. PhD thesis, Department of Computer Science, University College London, 1998.

[AMK97] S Abbasi, F Mokhtarian, and J Kittler. Reliable classification of chrysanthemum leaves through curvature scale space. In B ter Haar Romany, Luc Florack, Jan Koenderink, and Max Viergever, editors, *Scale-Space Theory in Computer Vision*, Lecture Notes in Computer Science No. 1252. Springer, 1997.

[AMO99] H J Andersen, J A Marchant, and C M Onyango. Evaluation of an imaging sensor for detecting vegetation using different waveband combinations. *International Journal of Imaging Systems Technology*, 1999. Submitted.

[AOM99] H J Andersen, C M Onyango, and J A Marchant. The design and operation of an imaging sensor for detecting vegetation. *International Journal of Imaging Systems Technology*, 1999. Submitted.

[Bau96] A Baumberg. Hierarchical shape fitting using an iterated linear filter. In *Proceedings of the British Machine Vision Conference*, volume 1, pages 313–322, 1996.

[BCZ93] A Blake, R Curwen, and A Zisserman. A framework for spatio-temporal control in the tracking of visual contours. *International Journal of Computer Vision*, 11(2):127–145, 1993.

[BDWH+90] M Brady, H Durrant-Whyte, H Hu, P Probert, and B S Y Rao. Sensor-based control of AGVs. *Computing and Control Engineering Journal*, pages 64–70, 1990.

[BH97] R G Brown and P W C Hwang. *Introduction to random signals and applied Kalman filtering*. John Wiley and Sons, 3rd edition, 1997.

[Bil98] H.R. Biller. Reduced input of herbicides by use of optoelectronic sensors. *Journal of Agricultural Engineering Research*, 71:357–362, 1998.

[BIR94] A Blake, M Isard, and D Reynard. Learning to track curves in motion. In *Proc. IEEE Int. Conf. Decision Theory and Control*, pages 3788–3793, 1994.

[BLD+98] S Baten, M Lützeler, E D Dickmanns, R Mandelbaum, and P J Burt. Techniques for autonomous, off-road navigation. *IEEE Intelligent Systems*, 1998.

[BM96] R Brivot and J A Marchant. Segmentation of plants and weeds for a precision crop protection robot using infrared images. *IEE Proc.-Vis. Image Signal Process.*, 143(2), April 1996.

[BPT98] P H Batavia, D A Pomerleau, and C E Thorpe. Predicting lane position for roadway departure prevention. In *Proceedings of the IEEE Intelligent Vehicles Symposium '98*, 1998.

[BS97] J Billingsley and M Schoenfisch. The successful development of a vision guidance system for agriculture. *Computers and Electronics in Agriculture*, 16:147–163, 1997.

[BSF88] Y Bar-Shalom and T Fortmann. *Tracking and Data Association*. Academic Press, New York, 1988.

[CB92] R M Curwen and A Blake. Dynamic contours: Real-time active splines. In A Blake and A Yuille, editors, *Active Vision*, chapter 3. MIT Press, 1992.

[CB98] N Chatterjee and B F Buxton. Shape modelling for drivable open terrains: Some studies. In *Proceedings of the Indian Conference on Computer Vision, Graphics and Image Processing (ICVGIP-98)*, December 1998.

[CDW94] S Cooper and H Durrant-Whyte. A Kalman filter for GPS navigation of land vehicles. In *Proceedings IEEE/RSJ International Workshop on Intelligent Robots and Systems, IROS '94*, pages 157–163, 1994.

[Coh91] L D Cohen. Note: On active contour models and balloons. *CVGIP (Image Understanding)*, 53(2):211–218, 1991.

[CP94] S Cameron and P J Probert, editors. *Advanced Guided Vehicles: Aspects of the Oxford AGV Project*. World Scientific, 1994.

[Cro89] J L Crowley. World modelling and position estimation for a mobile robot using ultrasonic ranging. In *IEEE International Conference on Robotics and Automation (ICRA89)*, pages 674–681, 1989.

[Cro95] J L Crowley. Mathematical foundations of navigation and perception for an autonomous mobile robot. In *Workshop on Reasoning with Uncertainty in Robotics*, December 1995.

[CT92] T F Cootes and C H Taylor. Active shape models – 'smart snakes'. In *Proceedings BMVC*, pages 266 – 275, 1992.

[CTCG92] T F Cootes, C J Taylor, D H Cooper, and J Graham. Training models of shape from sets of examples. In D Hogg and R Boyle, editors, *Proceedings BMVC*, pages 9–18. Springer Verlag, 1992.

[Dav98] A Davison. *Mobile robot navigation using active vision*. PhD thesis, Department of Engineering Science, University of Oxford, 1998.

[DF90] R Deriche and O Faugeras. Tracking line segments. *Image and Vision Computing*, 8(4):261 – 270, November 1990.

[DFBT99] Frank Dellaert, Dieter Fox, Wolfram Burgard, and Sebastian Thrun. Monte Carlo localization for mobile robots. In *IEEE International Conference on Robotics and Automation (ICRA99)*, May 1999.

[Dic98] E D Dickmanns. Vehicles capable of dynamic vision: a new breed of technical beings? *Artificial Intelligence*, 103:49–76, 1998.

[DM96] E D Dickmanns and N Müller. Scene recognition and navigation capabilities for lane changes and turns in vision-based vehicle guidance. *Control Engineering Practice*, 4(5):589–599, 1996.

[DW96] H F Durrant-Whyte. An autonomous guided vehicle for cargo handling applications. *The International Journal of Robotics Research*, 15(5):407–440, 1996.

[Fre61] H Freeman. On the encoding of arbitrary geometric configurations. *IEEE Trans. Elec. Computers*, EC-10:260–268, 1961.

[FSC98] G D Finlayson, B Schiele, and J L Crowley. Comprehensive colour image normalization. In *Computer Vision – ECCV '98*, pages 475–490, 1998.

[FWSB95] J M Ferryman, A D Worrall, G D Sullivan, and K D Baker. A generic deformable model for vehicle recognition. In *Proceedings BMVC*, volume 1, pages 127 – 136, 1995.

[Gop84] M Gopal. *Modern Control System Theory*. Wiley Eastern Ltd., 1984.

[GS66] D M Green and J A Swets. *Signal Detection Theory and Psychophysics*. John Wiley and Sons, 1966.

[Hag98] T Hague. Personal communication. Thresholding for plant matter extraction, 1998.

[Hag99] T Hague. Personal communication. Local mapping for crop treatment, 1999.

[Har92a] C Harris. Geometry from visual motion. In A Blake and A Yuille, editors, *Active Vision*, Artificial Intelligence, chapter 16, pages 263–284. MIT Press, 1992.

[Har92b] C Harris. Tracking with rigid models. In A Blake and A Yuille, editors, *Active Vision*, chapter 4. MIT Press, 1992.

[Hea95] A J Heap. Real-time hand tracking and gesture recognition using smart snakes. In *Proc. Interface to Real and Virtual Worlds*, Montpellier, June 1995.

[HGB97] H Hu, D Gu, and M Brady. Navigation and guidance of an intelligent mobile robot. In *2nd EUROMICRO workshop on Advanced Mobile Robots*, 1997.

[HJA+95] B Hosgood, S Jacquemoud, G Andreoli, J Verdebout, G Pedrini, and G Schmuck. Leaf optical properties experiment 93 (LOPEX93). Technical Report EUR 16095 EN, European Commission, Joint Research Centre, Institute for Remote Sensing Applications, 1995.

[HM82] J A Hanley and B J McNeil. The meaning and use of the area under a receiver operating characteristic (ROC) curve. *Radiology*, 143:29–36, April 1982.

[HMT97a] T Hague, J A Marchant, and N D Tillett. Autonomous robot navigation for precision horticulture. In *IEEE International Conference on Robotics and Automation*. Albuquerque, 1997.

[HMT97b] T Hague, J A Marchant, and N D Tillett. A system for plant scale husbandry. In *First European Conference on Precision Agriculture*, pages 635–642, 1997.

[HRL88] H R Hashemipour, S Roy, and A J Laub. Decentralized structures for parallel Kalman filtering. *IEEE Trans. Auto. Control*, 33(1):88–93, 1988.

[HS88] C G Harris and M J Stephens. A combined corner and edge detector. In *Proceedings of the* 4^{th} *Alvey Vision Conference*, pages 127–152, 1988.

[HT96] T Hague and N D Tillett. Navigation and control of an autonomous horticultural robot. *Mechatronics*, 6(2):165–180, 1996.

[IB96] M Isard and A Blake. Contour tracking by stochastic propagation of conditional density. In B Buxton and R Cipolla, editors, *Computer Vision – ECCV '96*, Lecture Notes in Computer Science. Springer, April 1996.

[IP94] J Ivins and J Porrill. Statistical snakes: Active region models. In *Proceedings BMVC*, volume 2, pages 377 – 386, 1994.

[Jaz70] A H Jazwinski. *Stochastic Process and filtering theory*. Academic Press, 1970.

[JZF98] M Jethwa, A Zisserman, and A Fitzgibbon. Real-time panoramic mosaics and augmented reality. In *British Machine Vision Conference*, pages 852–862, 1998.

[Kal60] R E Kalman. A new approach to linear filtering and prediction problems. *Transactions of the ASME Journal of Basic Engineering*, 1960.

[KB61] R E Kalman and R S Bucy. New results in linear filtering and prediction. *Trans. ASME, J. Basic Engineering*, 83:95–108, 1961.

[KWT87] M Kass, A Witkin, and D Terzopoulos. Snakes: Active contour models. In *First International Conference on Computer Vision*, pages 259 – 268, 1987.

[LDW91a] J J Leonard and H F Durrant-Whyte. Mobile robot localisation by tracking geometric beacons. *IEEE Trans. Robotics and Automation*, 7(3):376–382, June 1991.

[LDW91b] J J Leonard and H F Durrant-Whyte. Simultaneous map-building and localisation for an autonomous mobile robot. In *Proceedings IEEE/RSJ International Workshop on Intelligent Robots and Systems, IROS '91*, pages 1442–1447, 1991.

[LLF99] Y G Leclerc, Q-T Luan, and P Fua. Self-consistency: A novel approach to characterizing the accuracy and reliability of point correspondence algorithms. In A F Clark and P Courtney, editors, *ICVS Wrokshop on the Performance Characterisation and Benchmarking of Vision systems*, pages 1–19. BMVA, January 1999.

[Low90] D G Lowe. Stabilised solution for 3D model parameters. In *Proc. European Conference on Computer Vision*, pages 408–412, 1990.

[LRH94] D Langer, K Rosenblatt, and M Hebert. An integrated system for autonomous off-road navigation. In *Proceedings of the IEEE International Conference on Robotics and Automation '94*, 1994.

[MAD99] J R Miller, O Amidi, and M Delouis. Arctic test flights of the CMU autonomous helicopter. In *Proceedings of the Association for Unmanned Vehicle Systems International*, 1999.

[Mar96] J A Marchant. Tracking of row structure in three crops using image analysis. *Computers and electronics in agriculture*, 15(9):161–179, 1996.

[Mar98] J A Marchant. A measure of image quality suitable for acquisition control. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1998. Submitted December 1998.

[Mar99] J A Marchant. Personal communication. On the instability of crop/weed discrimination, 1999.

[Mas98] I Masaki. Machine-vision systems for intelligent transportation systems. *IEEE Intelligent Systems*, 1998.

[May79] P S Maybeck. *Stochastic Models, Estimation, and Control*, volume 1. Academic Press, 1979.

[May90] S J Maybank. Filter based estimates of depth. In *Proceedings BMVC*, pages 349–354, 1990.

[MB95] J A Marchant and R Brivot. Real time tracking of plant rows using a Hough transform. *Real Time Imaging*, 1:363–371, 1995.

[MF96] J F McLellan and L Friesen. Who needs a 20 cm precision farming system? In *Position Location and Navigation Symposium*, pages 426–432. IEEE, 1996.

[MM97] P F McLauchlan and J Malik. Vision for longitudinal vehicle control. In *Proceedings of the 8th British Machine Vision Conference*, volume 1, pages 260–269, 1997.

[MS97] J Matijevic and D Shirley. The mission and operation of the Mars Pathfinder microrover. *Control Engineering Practice*, 5(6):827–835, 1997.

[MSAW99] S Moorehead, R Simmons, D Apostolopoulos, and W Whittaker. Autonomous navigation field results of a planetary analog robot in Antartica. In *International symposium on AI, Robotics and Automation in Space '99*, 1999.

[MT97] N J B McFarlane and R D Tillett. Fitting 3D point distribution models of fish to stereo images. In *Proceedings BMVC*, volume 1, pages 330–339, 1997.

[NB97] B North and A Blake. Using expectation-maximisation to learn dynamical models from visual data. In *Proceedings of the British Machine Vision Conference*, pages 669–678, 1997.

[NDW99] E M Nebot and H Durrant-Whyte. A high integrity navigation architecture for outdoor autonomous vehicles. *Robotics and Autonomous Systems*, 26:81–97, 1999.

[Nie94] H M Nielsen. Modelling plant leaf shape for plant recognition. In Z S Chalabi, W Day, and P C Young, editors, *Proc. Second IFAC/ISHS Workshop on Mathematical and Control Applications in Agriculture and Horticulture*, Acta Horticulturae. ISHS, 1994.

[Nil69] N J Nilsson. A mobile automaton: an application of artificial intelligence techniques. In *1st International Joint Conference on Artificial Intelligence*, pages 509–520, 1969.

[OB97] J Ostrowski and J Burdick. Controllability tests for mechanical systems with constraints and symmetries. *Journal of Applied Mathematics and Computer Science*, 7(2):305–331, 1997.

[ON95] M Oren and S K Nayar. Generalization of the Lambertian model and implications for machine vision. *International Journal of Computer Vision*, 14:227–251, 1995.

[OS97] M Ollis and A Stentz. Vision-based perception for an automated harvester. In *Proceedings IEEE/RSJ International Workshop on Intelligent Robots and Systems, IROS '97*, volume 3, pages 1838–1844, 1997.

[PDK96] P L Palmer, H Dabis, and J Kittler. A performance measure for boundary detection algorithms. *Computer Vision and Image Understanding*, 63(3):474–494, 1996.

[PPP+89] S B Pollard, T P Pridmore, J Porrill, J E W Mayhew, and J P Frisby. Geometrical modelling from multiple stereo views. *International Journal of Robotics Research*, 8(4):132–138, 1989.

[Pra91] W K Pratt. *Digital image processing*. John Wiley, second edition, 1991.

[PSM97] F Pla, J M Sanchiz, and J A Marchant. Using perspective information to guide a row crop navigation vehicle. *Image and Vision Computing*, 15(6):465–474, 1997.

[Rao92] B Rao. Data association methods for tracking systems. In A Blake and A Yuille, editors, *Active Vision*, chapter 6. MIT Press, 1992.

[RDWS93] B S Y Rao, H F Durrant-Whyte, and J A Sheen. A fully decentralized multi-sensor system for tracking and surveillance. *International Journal of Robotics Research*, 12(1):20–44, February 1993.

[Ren93] W D Rencken. Concurrent localisation and map building for mobile robots using ultrasonic sensors. In *IEEE/RSJ International Conference on Intelligent Robots and Systems 1993*. IEEE, 1993.

[RWBM96] D Reynard, A Wildenberg, A Blake, and J A Marchant. Learning dynamics of complex motions from image sequences. In B Buxton and R Cipolla, editors, *Computer Vision – ECCV '96*, Lecture Notes in Computer Science. Springer, April 1996.

[SBIM99] J Sullivan, A Blake, M Isard, and J MacCormick. Object localization by Bayesian correlation. In *Proc. International Conference on Computer Vision*, 1999. In Press.

[SBM98] B Southall, B F Buxton, and J A Marchant. Controllability and observability: Tools for Kalman filter design. In M S Nixon, editor, *Proceedings 9^{th} British Machine Vision Conference*, volume 1, pages 164–173, September 1998.

[SBT97] N Sumpter, R D Boyle, and R D Tillett. Modelling collective animal behaviour using extended point distribution models. In *Proceedings BMVC*, volume 1, pages 242–251, 1997.

[SGH+97] Reid Simmons, R. Goodwin, K. Haigh, S. Koenig, and J. Sullivan. A layered architecture for office delivery robots. In *First International Conference on Autonomous Agents*, pages 235 – 242, February 1997.

[SH95] A Stentz and M Hebert. A complete navigation system for goal acquisition in unknown environments. *Autonomous Robots*, 2(2), 1995.

[SHMB99] B Southall, T Hague, J A Marchant, and B F Buxton. Vision-aided outdoor navigation of an autonomous horticultural vehicle. In Henrik I Christensen, editor, *Proceedings 1^{st} International Conference on Vision Systems*. Springer Verlag, January 1999.

[SMHB98] B Southall, J A Marchant, T Hague, and B F Buxton. Model based tracking for navigation and segmentation. In H Burkhardt and B Neumann, editors, *Proceedings 5^{th} European Conference on Computer Vision*, volume 1, pages 797–811, June 1998.

[Smi95a] S M Smith. ALTRUISM: Interpretation of three-dimensional information for autonomous vehicle control. *Engineering Applications of Artificial Intelligence*, 8(3):271–280, 1995.

[Smi95b] S M Smith. ASSET-2: Real-time motion segmentation and object tracking. Technical Report TR95SMS2b, Oxford Centre for Magnetic Resonance Imaging of the Brain, 1995.

[SNP98] M J J Scott, M Niranjan, and R W Prager. Realisable classifiers: Improving operating performance on variable cost problems. In *Proceedings BMVC 1998*, volume I, pages 306–315, September 1998.

[Soi99] P. Soille. Morphological image analysis applied to crop field mapping. *Image Vision and Computing*, 1999. In Preparation.

[Spi80] M R Spiegel. *Probability and Statistics*. Schaum's Outline Series. McGraw Hill, 1980.

[SPMB96] J M Sanchiz, F Pla, J A Marchant, and R Brivot. Structure from motion techniques applied to crop field mapping. *Image and Vision Computing*, 14:353–363, 1996.

[SSDW95] A Stevens, M Stevens, and H F Durrant-Whyte. OXNAV: Reliable autonomous navigation. In *Proceedings of the 1995 IEEE International Conference on Robotics and Automation*, volume 3, pages 2607–2612, 1995.

[Sta96] J V Stafford. Spatially variable field operations. *Computers and Electronics in Agriculture Special Issue on Spatially Variable Field Operations*, 14(2–3):99–100, 1996.

[Ste89] R S Stephens. Real-time 3D object tracking. In *Proceedings of the Fifth Alvey Vision Conference*, pages 85–90, 1989.

[SWB92] L S Shapiro, H Wang, and J M Brady. A matching and tracking strategy for independently moving objects. In *Proceedings BMVC*, pages 306 – 315, 1992.

[SWHB90] G D Sullivan, A D Worrall, R W Hockney, and K D Baker. Active contours in medical image processing using a networked SIMD array processor. In *First BMVC*, pages 395–400, 1990.

[THKS88] C Thorpe, M H Hebert, T Kanade, and S A Shafer. Vision and navigation for the Carnegie-Mellon Navlab. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 10(3), 1988.

[THM96] N D Tillett, T Hague, and J A Marchant. Plant scale crop protection. *Pesticide Outlook*, pages 25–26, August 1996.

[TJP97] C Thorpe, T Jochem, and D Pomerleau. Automated highways and the free agent demonstration. In *International Symposium on Robotics Research*, 1997.

[TMGM88] M A Turk, D G Morgenthaler, K D Gremban, and M Marra. VITS – a vision system for autonomous land vehicle navigation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 10(3), 1988.

[TMH96] N D Tillett, J A Marchant, and T Hague. Autonomous plant scale crop protection. In *AgEng Madrid*, 1996.

[TOM97] R D Tillett, C M Onyango, and J A Marchant. Using model-based image processing to track animal movements. *Computers and electronics in Agriculture*, 17:249–261, 1997.

[TS92] D Terzopoulos and R Szeliski. Tracking with Kalman snakes. In A Blake and A Yuille, editors, *Active Vision*, chapter 3. MIT Press, 1992.

[Tsa86] R Y Tsai. An efficient and accurate camera calibration technique for 3D machine vision. In *Proc. IEEE Computer Soc. Conf. Comp Vis Patt Recog*, Miami Beach, June 1986.

[Tsu94] S Tsugawa. Vision-based vehicles in Japan: Machine vision systems and driving control systems. *IEEE Transactions on Industrial Electronics*, 41(4), 1994.

[TZ97] P H S Torr and A Zisserman. Performance characterization of fundamental matrix estimation under image degradation. *Machine Vision and Applications*, 9:321–333, 1997.

[vBSK98] B van Branniken, M Stavridi, and J J Koenderink. Diffuse and specular reflectance from rough surfaces. *Applied Optics*, 37(1):130–139, January 1998.

[vdW69] B L van der Waerden. *Mathematical Statistics*. George Allen & Unwin Ltd., 1969.

[vT68] H L van Trees. *Detection, Estimation and Modulation Theory, Part I*. John Wiley and Sons, 1968.

[WCD76] D Willner, C B Chang, and K P Dunn. Kalman filter algorithms for a multi-sensor system. In *Proc. IEEE Int. Conf. Decision and Control*, pages 570–574, 1976.

[Wil97] A P Wildenberg. *Learning and Initialisation for Visual Tracking*. PhD thesis, University of Oxford Dept. Engineering Science, Oxford, UK, 1997.

[WNDDW00] S B Williams, P Newman, G Dissanayake, and H Durrant-Whyte. Autonomous underwater simultaneous localisation and map building. In *IEEE International conference on robotics and automation*, 2000. submitted.

[WOFH93] M Waite, M Orr, R Fisher, and J Hallam. Statistical partial constraints for 3D model matching and pose estimation problems. In *Proc. British Machine Vision Conference*, pages 105–114, 1993.

[YCH89] A L Yuille, D S Cohen, and P W Hallinan. Feature extraction from faces using deformable templates. In *Proc. Computer Vision and Pattern Recognition*, pages 104–110, 1989.

[YOI88] J Yoshida, S Okuyama, and K Ito. Automatic control of agricultural machines. In *Proceedings of the International Congress On Transportation Electronics '88*, 1988.

[Zha97] Z Y Zhang. Parameter estimation techniques: a tutorial with respect to conic fitting. *Image and Vision Computing*, 15(1):59–76, 1997.

[ZLY95] S C Zhu, T S Lee, and A L Yuille. Region competition: Unifying snakes, region growing, energy/Bayes/MDL for multi-band image segmentation. In *Proc. Fifth International Conference on Computer Vision*, pages 416–423, 1995.

[Zwi98] R Zwiggelaar. A review of spectral properties of plants and their potential use for crop/weed discrimination in row-crops. *Crop Protection*, 17(3):189–206, 1998.

Appendix A

The Hough transform row tracker

Prior to the development of the computer vision algorithms presented in this thesis, the autonomous vehicle's vision system used a Hough transform algorithm [Pra91] devised by Marchant and Brivot [MB95] to locate the crop row structure in each image. The Hough transform is a robust method for finding structures in images where the object of interest is represented by a parameterised model, and image features offer support to certain parameter values in a voting scheme. The set of parameters which receive most support are deemed to represent the object's state. We now present the derivation of the Hough transform given by Marchant and Brivot [MB95].

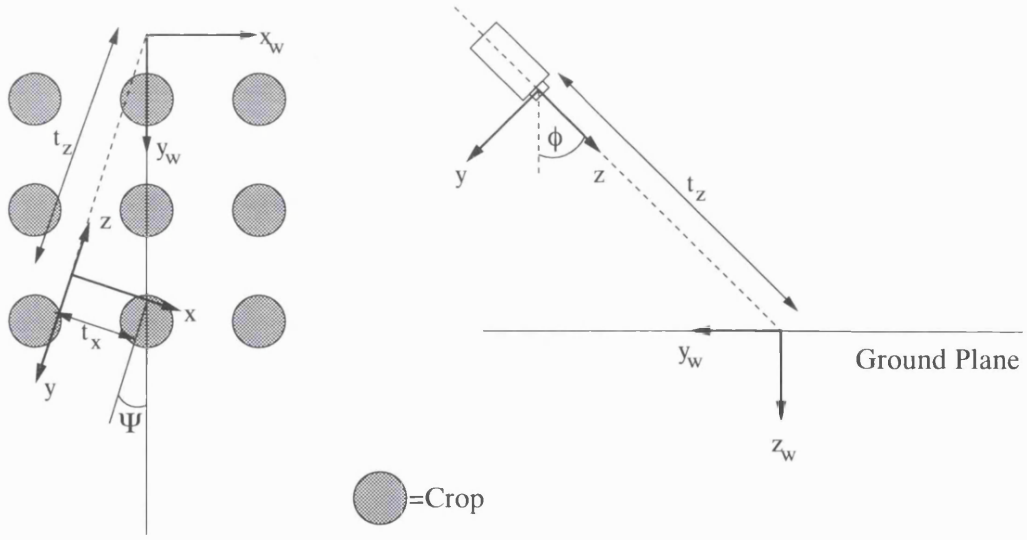


Figure A.1: The camera and world plane co-ordinate systems, with the rows of crop marked as grey circles.

Figure A.1 shows the co-ordinate systems of the vehicle camera (x, y, z) and the world (x_w, y_w, z_w), and figure A.2 shows an example image captured from the vehicle, and the blob features extracted by grey-level thresholding. Each blob is presented to the algorithm in terms of

its centroid in pixel co-ordinates x_f, y_f and its size calculated by the number of pixels it contains.

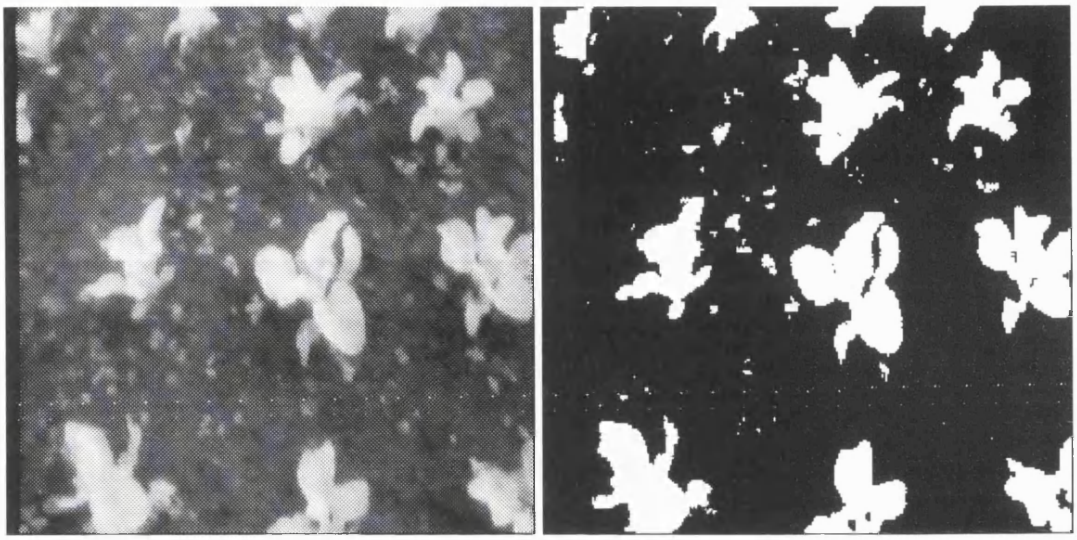


Figure A.2: The crop as seen by the camera. Left: original infra-red image. Right: thresholded version.

Tsai [Tsa86] gives the following transformation between the co-ordinate systems drawn:

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} \cos\Psi & \sin\Psi & 0 \\ -\sin\Psi\cos\phi & \cos\Psi\cos\phi & \sin\phi \\ \sin\Psi\sin\phi & -\cos\Psi\sin\phi & \cos\phi \end{bmatrix} \begin{bmatrix} x_w \\ y_w \\ z_w \end{bmatrix} + \begin{bmatrix} t_x \\ 0 \\ t_z \end{bmatrix}. \tag{A.1}$$

If we assume a pinhole camera model with focal length f, we obtain the following equations for the image plane co-ordinate x_u, y_u

$$x_u = f\frac{x}{z} = f\frac{(x_w\cos\Psi + y_w\sin\Psi + t_x)}{x_w\sin\Psi\sin\phi - y_w\cos\Psi\sin\phi + t_z}, \tag{A.2}$$

$$y_u = f\frac{y}{z} = f\frac{(-x_w\sin\Psi\cos\phi + y_w\cos\Psi\cos\phi + t_y)}{x_w\sin\Psi\sin\phi - y_w\cos\Psi\sin\phi + t_z}. \tag{A.3}$$

With reference to figure A.1, we can see that the crop form regular rows parallel to the y_w axis. The x_w co-ordinate of each row is given by

$$x_w = nr, \tag{A.4}$$

where $n \in -1, 0, 1$ and r is the distance between rows. If we assume that Ψ is small such that $\cos\Psi \approx 1$ and $\sin\Psi \approx \Psi$, then we can obtain the following relationships by manipulating equations A.2 and A.3, and substituting for x_w according to equation A.4:

$$y_w(-x_u\sin\phi - f\Psi) = -x_u(nr\sin\phi + t_z) + f(nr + t_x), \tag{A.5}$$

$$y_w(-y_u \sin\phi - f\cos\phi) = -y_u(\Psi nr \sin\phi + t_z) + f(-nr\cos\phi). \quad \text{(A.6)}$$

If we now eliminate y_w and simplify, we get

$$A\Psi^2 + D\Psi + E(t_x + nr) + F = 0, \quad \text{(A.7)}$$

where $A = nr(y_u \sin\phi + f\cos\phi)$, $D = y_u t_z$, $E = y_u \sin\phi + f\cos\phi$, and $F = -x_u t_z \cos\phi$.

For each feature, whose centre is x_f, y_f, we can obtain values for x_u and y_u thus:

$$x_u = (x_f - C_x)dx, \quad \text{(A.8)}$$

$$y_u = (y_f - C_y)dy, \quad \text{(A.9)}$$

where C_x and C_y are the pixel co-ordinates of the camera's optic axis and d_x and d_y are scaling factors determined by the dimensions of the pixels.

With x_u, y_u determined for a feature x_f, y_f, the only unknowns in equation A.7 are t_x and Ψ. The solutions of equation A.7 for the feature x_f, y_f give all the values of t_x and Ψ that could explain the presence of the feature at that particular position.

If we regard the two-dimensional plane of t_x and Ψ as a Hough (parameter) space, then we can map feature points x_f, y_f to curves in the Hough Space, using equations A.8,A.9 and A.7. Each feature produces three curves in the Hough space, corresponding to $n = -1$, $n = 0$ and $n = 1$. Each time a curve is plotted into the Hough space accumulator, it increments each accumulator cell it passes through by an amount proportional to the area of the feature that produced the curve. When the three curves corresponding to every feature in the image have been added to the Hough space, the accumulator cell with the highest value gives the t_x and Ψ values that best fit the image features. The Hough accumulator space and the resulting row fitting for the image on the left of figure A.2 are given in figure A.3.

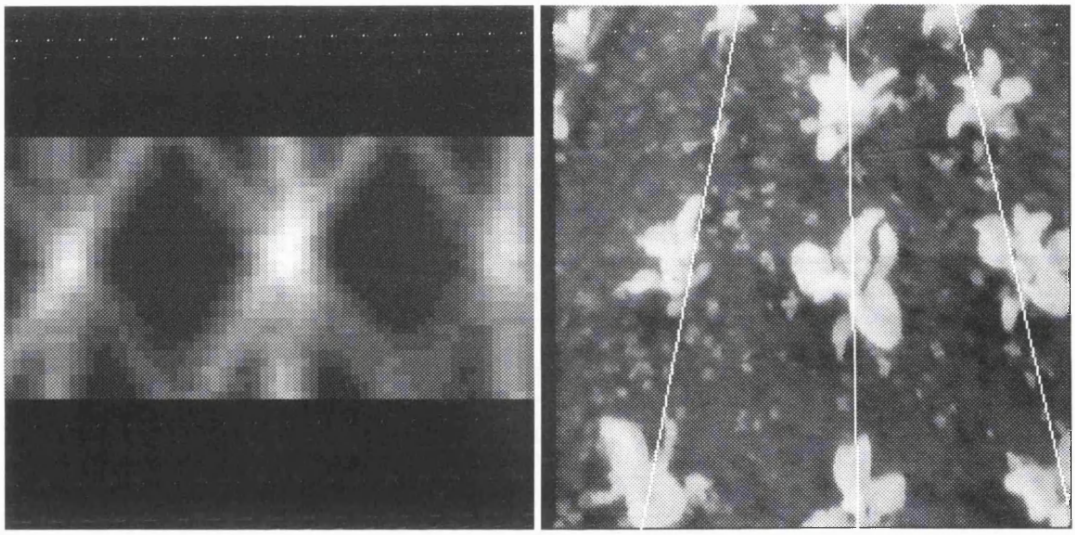


Figure A.3: The Hough accumulator and the fitted row structure. Left: the Hough accumulator space – the lighter the pixel, the higher the accumulator count. Right: the row structure position corresponding to the accumulator cell with the highest count superimposed on the original image from figure A.2.

Appendix B

The matrix inversion lemma

The matrix inversion lemma states that given the matrices $\mathbf{A}, \mathbf{B}, \mathbf{C}$ and $\mathbf{D}$, the sum of $\mathbf{A}$ with the product $\mathbf{BCD}$ may be inverted as follows:

$$[\mathbf{A} + \mathbf{BCD}]^{-1} = \mathbf{A}^{-1} - \mathbf{A}^{-1}\mathbf{B}[\mathbf{C}^{-1} + \mathbf{DA}^{-1}\mathbf{B}]^{-1}\mathbf{DA}^{-1}. \tag{B.1}$$

Appendix C

Conditional density formulation of the Kalman filter

Given the conditional density formulation for the Kalman filter, with Gaussian noise sources, the following expression has been arrived at in equation 3.22 (chapter 3), repeated here for clarity:

$$f((\mathbf{x}(k)|\mathbf{z}(k)) = \frac{N(\mathbf{H}(k)\mathbf{x}(k), \mathbf{R}(k))N(\hat{\mathbf{x}}^-(k), \mathbf{P}^-(k))}{N(\mathbf{H}(k)\mathbf{x}^-(k), \mathbf{H}(k)\mathbf{P}^-(k)\mathbf{H}^T(k) + \mathbf{R}(k))}. \tag{C.1}$$

If we expand the normal distributions $N(\cdot,\cdot)$, equation C.1 reduces to a normal distribution whose mean and variance will be arrived by manipulation of the exponential term of equation C.1, which is given by

$$\begin{aligned} E = (\mathbf{z} - \mathbf{H}\mathbf{x})^T\mathbf{R}^{-1}(\mathbf{z} - \mathbf{H}\mathbf{x}) + (\mathbf{x} - \hat{\mathbf{x}}^-)^T(\mathbf{P}^-)^-1(\mathbf{x} - \hat{\mathbf{x}^-}) - \\ (\mathbf{z} - \mathbf{H}\hat{\mathbf{x}}^-)^T(\mathbf{H}\mathbf{P}^-\mathbf{H}^T + \mathbf{R})^-1(\mathbf{z} - \mathbf{H}\hat{\mathbf{x}}^-), \end{aligned} \tag{C.2}$$

where the time indices have been dropped for brevity. Upon expansion of the quadratic expressions involving $\mathbf{x}$, equation C.2 becomes

$$\begin{aligned} E = \mathbf{z}^T\mathbf{R}^{-1}\mathbf{z} - \mathbf{z}^T\mathbf{R}^{-1}\mathbf{H}\mathbf{x} - \mathbf{x}^T\mathbf{H}^T\mathbf{R}^{-1}\mathbf{z} + \mathbf{x}^T\mathbf{H}^T\mathbf{R}^{-1}\mathbf{H}\mathbf{x} + \\ \mathbf{x}^T(\mathbf{P}^-)^{-1}\mathbf{x} - \mathbf{x}(\mathbf{P}^-)^{-1}\hat{\mathbf{x}}^- - (\hat{\mathbf{x}}^-)^T(\mathbf{P}^-)^{-1}\mathbf{x} + \hat{\mathbf{x}}^-(\mathbf{P}^-)^{-1}\hat{\mathbf{x}}^- - \\ (\mathbf{z} - \mathbf{H}\hat{\mathbf{x}}^-)^T(\mathbf{H}\mathbf{P}^-\mathbf{H}^T + \mathbf{R})^-1(\mathbf{z} - \mathbf{H}\hat{\mathbf{x}}^-). \end{aligned} \tag{C.3}$$

If we now complete the square in $\mathbf{x}$, we arrive at the following expression for the exponent of equation C.1;

$$\begin{aligned} E = (\mathbf{x} - \mathbf{P}[\mathbf{H}^T\mathbf{R}^{-1}\mathbf{z} + (\mathbf{P}^-)^{-1}\hat{\mathbf{x}}^-])^T\mathbf{P}^{-1}(\mathbf{x} - \mathbf{P}[\mathbf{H}^T\mathbf{R}^{-1}\mathbf{z} + (\mathbf{P}^-)^{-1}\hat{\mathbf{x}}^-]) - \\ ((\hat{\mathbf{x}}^-)^T(\mathbf{P}^-)^{-1} + \mathbf{z}^T\mathbf{R}^{-1}\mathbf{H})\mathbf{P}(\mathbf{H}^T\mathbf{R}^{-1}\mathbf{z} + (\mathbf{P}^-)^{-1}\hat{\mathbf{x}}^-) + \\ (\hat{\mathbf{x}}^-)^T(\mathbf{P}^-)^{-1}\hat{\mathbf{x}}^- + \mathbf{z}^T\mathbf{R}^{-1}\mathbf{z} - (\mathbf{z} - \mathbf{H}\hat{\mathbf{x}}^-)^T(\mathbf{H}\mathbf{P}^-\mathbf{H}^T + \mathbf{R})^-1(\mathbf{z} - \mathbf{H}\hat{\mathbf{x}}^-), \end{aligned} \tag{C.4}$$

where the matrix $\mathbf{P}$ is given by

$$\mathbf{P} = [\mathbf{H}^T\mathbf{R}^{-1}\mathbf{H} + (\mathbf{P}^-)^{-1}]^{-1}, \tag{C.5}$$

which is familiar from equation 3.24 in chapter 3.

Equation C.4 is comprised of two parts, a quadratic in $\mathbf{x}$ and a "remainder". If this remainder is zero, then the mean and covariance of the posterior density (and hence the updated state estimate and its covariance) can be derived from the quadratic form.

Denoting the remainder R, we aim to demonstrate that

$$R = ((\hat{\mathbf{x}}^-)^T(\mathbf{P}^-)^{-1} + \mathbf{z}^T\mathbf{R}^{-1}\mathbf{H})\mathbf{P}(\mathbf{H}^T\mathbf{R}^{-1}\mathbf{z} + (\mathbf{P}^-)^{-1}\hat{\mathbf{x}}^-) - (\hat{\mathbf{x}}^-)^T(\mathbf{P}^-)^{-1}\hat{\mathbf{x}}^- - \mathbf{z}^T\mathbf{R}^{-1}\mathbf{z} + (\mathbf{z} - \mathbf{H}\hat{\mathbf{x}}^-)^T(\mathbf{H}\mathbf{P}^-\mathbf{H}^T + \mathbf{R})^-1(\mathbf{z} - \mathbf{H}\hat{\mathbf{x}}^-) = 0, \tag{C.6}$$

or equivalently,

$$((\hat{\mathbf{x}}^-)^T(\mathbf{P}^-)^{-1} + \mathbf{z}^T\mathbf{R}^{-1}\mathbf{H})\mathbf{P}(\mathbf{H}^T\mathbf{R}^{-1}\mathbf{z} + (\mathbf{P}^-)^{-1}\hat{\mathbf{x}}^-) = (\hat{\mathbf{x}}^-)^T(\mathbf{P}^-)^{-1}\hat{\mathbf{x}}^- - \mathbf{z}^T\mathbf{R}^{-1}\mathbf{z} + (\mathbf{z} - \mathbf{H}\hat{\mathbf{x}}^-)^T(\mathbf{H}\mathbf{P}^-\mathbf{H}^T + \mathbf{R})^-1(\mathbf{z} - \mathbf{H}\hat{\mathbf{x}}^-). \tag{C.7}$$

If we expand the matrix quadratics on each side of equation C.7, whilst applying the matrix inversion lemma (appendix B) to the matrix $[\mathbf{H}\mathbf{P}\mathbf{H}^T + \mathbf{R}]^{-1}$, we arrive at the following equality (terms common to each side have been removed, and some re-arrangement performed):

$$\begin{aligned} &(\hat{\mathbf{x}}^-)^T[\mathbf{I} - \mathbf{H}^T\mathbf{R}^{-1}\mathbf{H}\mathbf{P}]\mathbf{H}^T\mathbf{R}^{-1}\mathbf{z} + \\ &\mathbf{z}^T\mathbf{R}^{-1}\mathbf{H}[\mathbf{I} - \mathbf{P}\mathbf{H}^T\mathbf{R}^{-1}\mathbf{H}]\hat{\mathbf{x}}^- + \\ &\hat{\mathbf{x}}^-[\mathbf{P}^{-^{-1}} - \mathbf{H}^T\mathbf{R}^{-1}\mathbf{H} + \mathbf{H}^T\mathbf{R}^{-1}\mathbf{H}\mathbf{P}\mathbf{H}^T\mathbf{R}^{-1}\mathbf{H}]\hat{\mathbf{x}}^- \end{aligned} \quad = \quad \begin{aligned} &\hat{\mathbf{x}}^-(\mathbf{P}^-)^{-1}\mathbf{P}\mathbf{H}^T\mathbf{R}^{-1}\mathbf{z} + \\ &\mathbf{z}^T\mathbf{R}^{-1}\mathbf{H}\mathbf{P}(\mathbf{P}^-)^{-1}\hat{\mathbf{x}}^- + \\ &\hat{\mathbf{x}}^-(\mathbf{P}^-)^{-1}\mathbf{P}(\mathbf{P}^-)^{-1}\hat{\mathbf{x}}^-, \end{aligned} \tag{C.8}$$

where $\mathbf{I}$ is the identity matrix. Equation C.8 may appear to be curiously arranged; this is in order to group the similar terms on each side. The proof of equality in equation C.7 can now be achieved by comparing terms from equation C.8; in fact, owing to the symmetry of the matrices involved, of the three lines in equation C.8, only two equalities need to be shown:

$$\mathbf{I} - \mathbf{H}^T\mathbf{R}^{-1}\mathbf{H}\mathbf{P} = (\mathbf{P}^-)^{-1}\mathbf{P}, \tag{C.9}$$

and

$$(\mathbf{P}^-)^{-1} - \mathbf{H}^T\mathbf{R}^{-1}\mathbf{H} + \mathbf{H}^T\mathbf{R}^{-1}\mathbf{H}\mathbf{P}\mathbf{H}^T\mathbf{R}^{-1}\mathbf{H} = (\mathbf{P}^-)^{-1}\mathbf{P}(\mathbf{P}^-)^{-1}. \tag{C.10}$$

The proof of equation C.9 follows directly from the definition of $\mathbf{P}$ in equation C.5, whilst equation C.10 may be proved using the result of C.9 and the symmetry of $(\mathbf{P}^-)^{-1}$ and $\mathbf{P}$.

Now that we have proved equation C.6, i.e. that the remainder term in the posterior density function is zero, the exponential term E can be written

$$E = (\mathbf{x} - \mathbf{P}[\mathbf{H}^T\mathbf{R}^{-1}\mathbf{z} + (\mathbf{P}^-)^{-1}\hat{\mathbf{x}}^-])^T\mathbf{P}^{-1}(\mathbf{x} - \mathbf{P}[\mathbf{H}^T\mathbf{R}^{-1}\mathbf{z} + (\mathbf{P}^-)^{-1}\hat{\mathbf{x}}^-]). \tag{C.11}$$

This exponential term leads to a distribution with mean

$$\mathbf{P}[\mathbf{H}^T\mathbf{R}^{-1}\mathbf{z} + (\mathbf{P}^-)^{-1}\hat{\mathbf{x}}^-], \tag{C.12}$$

where $\mathbf{P}$ is the matrix defined above in equation C.5 – this is also the covariance of the distribution specified by E. If we apply the matrix inversion lemma to $\mathbf{P}$ (equation C.5), and insert this into equation C.12, we get

$$[\mathbf{P}^- - \mathbf{P}^-\mathbf{H}^T[\mathbf{H}\mathbf{P}^-\mathbf{H}^T + \mathbf{R}]^{-1}\mathbf{H}\mathbf{P}^-][\mathbf{H}^T\mathbf{R}^{-1}\mathbf{z} + (\mathbf{P}^-)^{-1}\hat{\mathbf{x}}^-] = \hat{\mathbf{x}}^- + [\mathbf{I} - \mathbf{P}^-\mathbf{H}^T[\mathbf{H}\mathbf{P}^-\mathbf{H}^T + \mathbf{R}]^{-1}\mathbf{H}]\,\mathbf{P}^-\mathbf{H}^T\mathbf{R}^{-1}\mathbf{z} + \mathbf{P}^-\mathbf{H}^T[\mathbf{H}\mathbf{P}^-\mathbf{H}^T + \mathbf{R}]^{-1}\mathbf{H}\hat{\mathbf{x}}^-, \quad \text{(C.13)}$$

which in turn simplifies to yield (with time index re-inserted)

$$\hat{\mathbf{x}}(k) = \hat{\mathbf{x}}^-(k) + \mathbf{P}^-(k)\mathbf{H}^T(k)[\mathbf{H}(k)\mathbf{P}^-(k)\mathbf{H}^T(k) + \mathbf{R}(k)]^{-1}(\mathbf{z}(k) - \mathbf{H}(k)\hat{\mathbf{x}}^-(k)), \quad \text{(C.14)}$$

which is the Kalman filter state estimate update equation, as given in equation 3.23.