VISUAL TRACKING: FROM AN INDIVIDUAL TO GROUPS OF ANIMALS

Andrew Peter French, BSc.

Thesis submitted to the University of Nottingham
for the degree of Doctor of Philosophy

September 2005

Abstract

This thesis is concerned with the development and application of visual tracking techniques to the domain of animal monitoring. The development and evaluation of a system which uses image analysis to control the robotic placement of a sensor on the back of a feeding pig is presented first. This *single*-target monitoring application is then followed by the evaluation of suitable techniques for tracking *groups* of animals, of which the most suitable existing technique is found to be a Markov chain Monte Carlo particle filtering algorithm with a Markov random field motion prior (MCMC MRF, Khan et al. 2004). Finally, a new tracking technique is developed which uses social motion information present in groups of social targets to guide the tracking. This is used in the new Motion Parameter Sharing (MPS) algorithm.

MPS is designed to improve the tracking of groups of targets with coordinated motion by incorporating motion information from targets that have been moving in a similar way. Situations where coordinated motion information should improve tracking include animal flocking, people moving as a group or any situation where some targets are moving in a correlated fashion.

This new method is tested on a variety of real and artificial data sequences, and its performance compared to that of the MCMC MRF algorithm. The new MPS algorithm is found to outperform the MCMC MRF algorithm during a number of different types of sequences (including during occlusion events and noisy sequences) where correlated motion is present between targets. This improvement is apparent both in the accuracy of target location and robustness of tracking, the latter of which is greatly improved.

Published work

The following work from this thesis has been published:

French, A.P., Frost, A., Pridmore, T.P. and Tillett, R.D. 2003. An image analysis system to guide a sensor placement robot onto a feeding pig. *Proc. Irish Machine Vision and Image Processing Conference*, 185-192, 3rd-5th September 2003, Coleraine.

Frost, A., French, A.P., Tillett, R.D., Pridmore, T.P. and Welch, S.K. 2004. A vision guided robot for tracking a live, loosely constrained pig. *Computers and Electronics in Agriculture* **44**, 93-106.

Acknowledgements

I would like to thank my supervisors, Dr. Tony Pridmore and Dr. Robin Tillett, for their advice and help throughout my PhD; their enthusiasm and support was unwavering. Many thanks to everyone at Silsoe and Nottingham for being so great: to the sensing group, to anyone who helped with the animals, to Mum and Dad, and of course to all my family and friends.

And not forgetting my wife, Vicki; an unexpected but amazing outcome of this PhD.

Table of Contents

List of Figures

List of Tables

Chapter 1: Introduction

1.1. General Introduction

1.1.1. Objective and aim

This thesis presents the development and application of two novel visual tracking applications that are evaluated in real world monitoring situations. The hypothesis is that visually tracking animals using image analysis can provide a foundation for further monitoring tasks. These further tasks might require controlling some invasive sensing mechanism (Chapter 3) or analysing the motion of animals remotely allowing further monitoring of behaviour (Chapters 4 and 5). These areas are explored in this thesis by combining existing techniques in novel ways, and developing a new methodology for tracking groups of social targets.

Monitoring in this thesis is defined as the act of gaining information about an animal from the physical characteristics it exhibits during its normal routine. Such monitoring may be non-invasive monitoring, such as observing how an animal moves, or invasive monitoring, such as taking a measurement from an animal with a device that needs to be held in contact with the animal. In different situations, one method is normally more appropriate than the other. Approaches from both sides will be presented in this work.

It is the aim of this thesis, then, to develop image analysis techniques to be used as a basis for monitoring applications in the animal domain. A novel image analysis-controlled sensor placement system for locating a sensing position on a single

feeding pig is described first. Attention then turns to the monitoring, and hence visual tracking, of multiple targets. The strengths and weaknesses of two common tracking algorithms - Condensation (Isard and Blake 1998b) and Markov chain Monte Carlo (MCMC) particle filtering (Khan et al. 2004) - are demonstrated in the domain of multiple target tracking. A novel way of using social information to improve such tracking is then presented, in the form of the Motion Parameter Sharing (MPS) algorithm. Both the single and multiple animal tracking methods are implemented and tested in distinct real world environments. The potential of the new tracking stretches beyond the animal monitoring domain into any area where groups of targets need to be robustly tracked.

1.1.2. Motivation for the research

Much work exists which involves using CCTV-style surveillance setups to monitor pedestrians or cars in public areas (see Chapter 2 for a review of literature in this area). Animals are less commonly used as the subject of such monitoring in image analysis-based systems. This may be because people are more readily abundant subjects, and perhaps because it is easier to understand and predict the kind of motions that people exhibit (although this is still a major challenge, it is feasible to think that it is easier to understand someone's movements when you can ask them to explain their actions). Another reason is the special requirements that animals demand of the experimenter. Examples of such special requirements include the construction of specialist research areas (arenas in which to monitor the animals) and the obvious requirement to maintain the health of the animals being studied (feeding and cleaning). However, there are many reasons why it is important to be able to monitor animals; in fact there are perhaps more immediate uses for farm animal monitoring than for the monitoring of people. Unlike people, animals cannot verbally communicate their feelings, and so any information about their well-being must be gathered by observations or measurements. Access to this information can help improve the animals' welfare and increase the stockmen's profit. Taking these measurements or making observations is a very time-consuming and highly skilled operation; therefore, it has a substantial economic cost. Monitoring that could glean potentially valuable

information from animals is often carried out too infrequently to be of most use. Research in this field has then two justified and realistic goals: to save the animal husbandry industry money and to increase animal welfare quality.

Additionally, animals provide a tough test for the evaluation of image analysis algorithms. Animals are difficult targets to track using conventional image analysis techniques because of their inter-target similarity and hard-to-predict dynamics. This is compounded by an operating environment that can never be fully constrained. Though the same could be said of human targets, with animals these problems are intrinsic, whereas with people they are the exception to the rule: for example, people are often dressed differently and situations in which they are indistinguishable are comparatively less common. The environment in which people are tracked can often be manipulated to a greater extent than that of animals. Animal houses are the subject of numerous welfare laws and the economic constraints of the farmer, which together constrain the possible variations in factors such as lighting and background material properties.

One complex aspect which animal and human motion shares is the social aspect of interactions and social grouping: something which us as humans may not be aware of but which can be a driving force in movement. Consider a group of people walking to lunch together, or a group of friends waiting at a station, or confronting groups at a riot. These actions have social grouping components, which can be seen perhaps in a purer form in the animal kingdom with the likes of aggregations of animals hunting food, or the tight flocking effects present when a predator approaches. Therefore, once the challenges have been overcome in the animal domain, the techniques should be easy to convert to a more mainstream domain such as people tracking. Overcoming these challenges with animals opens the way to powerful and interesting new techniques for monitoring which, although developed for animals, could be adapted for use in many other areas, including the automated surveillance of human activities.

1.1.3. Overview of thesis

This thesis is organised as follows:

Chapter 2: A review of visual monitoring techniques and applications

Some common image analysis techniques are presented. A background to tracking and surveillance is given. Individual chapters have their own more specific literature reviews.

Chapter 3: Monitoring an animal by visual tracking: Automatic sensor positioning over the back of a feeding pig.

Describes the development of a system in which a single animal is to be monitored using a robotic sensor positioned by image analysis: a system is developed to locate a feeding pig, identify and locate feature points on the animal and direct a sensing robot arm to a specified point in relation to the feature points.

Chapter 4: Monitoring multiple animals by visual tracking: Video tracking of ducks in an outside arena

The performance of two powerful tracking algorithms is evaluated, along with an analysis of their shortcomings. The addition of an interaction-modelling motion model is shown to improve tracking, and multiple independent trackers are found to be unsatisfactory.

Chapter 5: Using social information to improve tracking performance for groups

The development of a new Motion Parameter Sharing technique that incorporates information about group behaviour into a tracking algorithm is described. This method allows correlated targets to be more accurately and robustly tracked in a variety of situations, compared to the most successful algorithm found in Chapter 4.

Chapter 6: General discussion and future work

Conclusions are drawn about the success of the tracking work and the achievements of this thesis, and a list of some possible future work and applications is presented

The potential of the new Motion Parameter Sharing technique is explored.

1.2. Main Contributions

There are two main contributions from this thesis. This first is the production of a novel system which is able to direct a robot arm to the P2 position on the back of a pig as it feeds. This is achieved by combining and tuning existing computer vision techniques to produce a novel application tested in a real-world domain.

The second contribution is the development of a novel, multiple-target tracking algorithm that can make use of the social motion information about the targets to help guide the tracking. This algorithm works by sharing motion parameters between targets that have been exhibiting coordinated motion; hence it is named the Motion Parameter Sharing algorithm.

The rest of this thesis describes how these contributions were achieved.

Chapter 2: A review of visual monitoring techniques and applications

2.1. Introduction

This chapter presents a general background review of some image analysis techniques that would typically be required for automated monitoring problems. This is followed by some of their example applications from the tracking literature. Specific literature related to particular chapters is reviewed in the chapters themselves; this chapter is intended as a background survey of the area and techniques in general, and to introduce the reader to relevant work in related fields. Therefore, the literature specifically relevant to monitoring an animal using image analysis to guide a sensing robot is presented in Chapter 3, and literature concerned with tracking and monitoring groups is presented in Chapter 4. The social tracking chapter (Chapter 5) reviews relevant background work regarding social effects in groups.

2.2. General literature review

2.2.1. Image analysis

Image analysis is the "process of discovering, identifying and understanding patterns that are relevant to the performance of an image-based task" (Gonzalez and Woods 1992). In other words, it is extracting useful information from image data, be it from a still image, stored video sequence or live camera source. Teaching a computer to 'see' is the ultimate goal. In layman's (or Aristotle's!)

terms, to see is to "know what is where by looking", and allowing a computer to make such judgements is a difficult and expansive task. Image analysis has been applied to a large number of problems in a diverse range of fields, from guiding machinery (Tillett and Hague 1999), controlling robots (Vaughan et al. 1998), industrial automation (Chin and Dyer 1986), medical imaging (McInerney and Terzopoulos 1996), and wildlife detection in aerial images (Sidle and Ziewitz 1990) to analysing social behaviour in nursing homes (Chen et al. 2004), and much more in between. All of the applications are a form of sensing, as to use a camera is to take measurements from a sensor that measures light. Image analysis as a foundation for monitoring interprets the measurements and infers knowledge about the location and motion of target objects (animals, in this case). This information can be further processed to identify higher-level events (Buxton 2003). Image analysis is a tool particularly suited to monitoring applications, as has been demonstrated in previous work. It allows non-invasive, remote sensing of targets, using equipment akin to that already existing in a vast and expanding CCTV network. Cameras are relatively cheap and extremely versatile, and can sense a wide area. Once in place, they can be used for a variety of software-driven tasks, e.g. detection of suspicious objects (Beynon et al. 2003) or tracking of pedestrian motion (Uchida et al. 2000). This non-invasive and adaptable nature makes image analysis very attractive for animal monitoring work.

2.2.2. Common image analysis techniques

An image analysis-driven monitoring system must perform a number of stages of processing, the essentials of which will now be presented. Depending on the quality of the captured image and the accuracies required, sensor noise might need to be attenuated, and lens distortion effects removed. The two major types of noise are impulsive noise ("salt and pepper noise") and Gaussian noise (Sonka et al. 1993). Gaussian noise may be removed by Gaussian smoothing, a specific instance of an averaging filter. The downside of this kind of filtering is blurring of the image (signal frequencies shared with the noise are lost) leading to poor feature localization; also it cannot effectively remove salt and pepper noise. Impulsive noise, caused perhaps by damaged CCD elements, can be removed by

median filtering. Median filtering is a non-linear technique in which each pixel value is replaced by the median value of all the pixels in the specified neighbourhood. Median filtering can completely suppress impulsive noise, and blurs contours less than averaging filters (Trucco and Verri 1998). As well as noise, image distortion can corrupt image data. There are two main components to image distortion caused by the capturing equipment. Radial or barrel distortion is the tendency of wider-angle lenses to pull points to the optical centre. Decentering distortions are caused by the non-orthogonality of lens components with respect to the optical axis, due typically to manufacturing inaccuracies (Conrady 1919). There exist a variety of methods to remove such effects: using known calibration points in space to recover distortion parameters (Tsai 1986), estimating parameters in order to map curved image lines back to straight lines (Swaminathan and Nayar 1998; Devernay and Faugeras 2001) and even with no straight lines or calibration information at all, by estimating distortion by looking for high-order correlations in the frequency domain (Farid and Popescu 2001).

The next stage after cleaning up the image is to locate the areas, features or objects of interest in the image, and this location is typically initiated using one or more of a series of low-level image processing operations. Thresholding is an image processing technique that selects all the pixels in an image that are above a certain intensity. Variations on it include thresholding with hysteresis, which selects all pixels above a certain value, *and* all pixels above a lower threshold that neighbour a higher-threshold point. Hysteresis thresholding is commonly used to detect lines from the output of an edge detector. Thresholding at its simplest can be used as the sole means of identifying targets in images, such as in bird censusing studies where the intensity difference between bird and background is large enough to warrant using this technique alone (Allen and Thorpe 1991). It is more typically used in conjunction with or to initialise other techniques, for example to help identify the boundary of a pig (Marchant and Schofield 1992; Marchant et al. 1999). Such boundaries are examples of features in images. Other features can include lines, edges, textures or in fact any part of the image with some special property. Feature detectors form the basis of many vision systems. One of the most fundamental features that is often required is the edge: a

step change in intensity. These are found at significant boundaries in the image, often between the foreground object of interest and the background. Once noise has been removed, there are two main steps in edge detection. First, a filter must be designed to give a large response at elements of the image where an edge exists, and a low response elsewhere. Second, from the output of the filter the edges must be located and noise suppressed. Some common early edge detectors include Sobel (Gonzalez and Woods 1992) and Robert's Cross (Roberts 1965). These are both based on gradient (first order derivative) measurements from the image, where a high response indicates an edge. It is also possible to find edges at zero crossings of second order derivatives; the Marr-Hildreth (Marr 1982) operator implements this approach. Using zero crossings always gives closed contours; though because a second derivative is calculated the operator is quite susceptible to noise. One of the most used edge detectors is the Canny edge detector (Canny 1986). This detector is optimal against certain specified criteria. The algorithm itself has a number of modules. First, the image is smoothed. Then a simple edge detector is applied to the image. It then tracks with hysterisis thresholding along the output 'ridges', setting to zero any pixels which are non-maximal. This gives 1-pixel wide lines as an output.

All the detectors above aim to locate edges or boundaries in images. An example of a more refined and statistically robust boundary locating technique is a 'Snake' (Kass et al. 1988), which is a form of active contour model. Contour models impose constraints on the shapes contours can adopt. With snakes, high-level constraints are imposed on the boundary geometry, balancing the attractive forces of apparent edges in the images, and the physical properties of the hypothesised contour itself. Snakes are useful for identifying boundaries that are easy for a human to detect, but would otherwise be difficult to detect using just low-level image processing techniques. They have been put to successful use in many situations, including traffic monitoring systems (e.g. Tai et al. 2004) and as a basis for dynamic contours (Blake and Isard 1998) used for tracking many different types of target.

As well as boundary detecting techniques, there are other methods that can be used to separate the area of interest from the background. Colour information is one common way of discriminating targets from background in surveillance and monitoring. The expected colour of a target area can be used as a measurement for a tracking algorithm (Nummiaro et al. 2003). A model of the target's colour is created from examples, and then a measure of the distance from this model describes how well a target's colour matches the target model. Segmentation is another method where areas of the image within certain colour parameters are identified in an image. Segmentation methods range from simple clustering of coloured pixels (for example using K-means clustering), through region and edge based methods (e.g. the Watershed algorithm (Vincent and Soille 1991)) to probabilistic methods using the EM Algorithm (Forsyth and Ponce 2003). Such segmentation techniques have been used for many applications, including aerial surveillance of wildlife (Sidle and Ziewitz 1990) and identifying football players (Vendenbroucke et al. 1998).

If a *sequence* of images is available, information in the time domain can be used to identify moving areas of the image. One example of this is optical flow, which uses the apparent flow of brightness features in the sequence to segment regions of motion from the background (Horn and Schunck 1981; Barron et al. 1994). Background subtraction is another method that can be used with sequences of images. This method builds a model of the background scene, which can then be subtracted from subsequent images of the same scene, leaving only the foreground objects of interest. Typically this needs an initial image of the scene containing no targets, but a background image can also be built up if all the foreground objects are moving and they cover each part of the background for a suitably small amount of time; for example median filtering requires that the background pixels be obscured for less than half of the number of frames. This background subtraction method has been used successfully in activity monitoring systems that track people (Grimson et al. 1998).

2.2.3. Review of major tracking techniques

One of the most famous tracking algorithms is the Kalman Filter (Kalman 1960). This algorithm is optimal in that it can incorporate all data available and give a best estimate (minimizing the error variance) as long as three criteria are met. These are: the system must be linear, and the measurement noise must be white and Gaussian (Welch and Bishop 2001). Although this algorithm has proved very popular, it has its disadvantages. One such restriction is that it is limited to unimodal Gaussian densities and so cannot represent multiple alternative hypotheses. This problem is resolved by another group of algorithms called particle filters. Particle filtering methods represent the state probability space as a set of particles. An example of one of these is Condensation (Isard and Blake 1998b). The Condensation method approximates a probability distribution with a set of samples using a method known as factored sampling. This kind of representation allows multi-modal, non-Gaussian state probability densities, the kind of which is common when clutter (competing, false measurements) is present in an image.

Tracking multiple targets is a specialised problem. Early major work in this area includes the Multiple Hypothesis Tracker (Reid 1979), which forms multiple target-data association hypotheses and calculates their probabilities. The Joint Probabilistic Data Association Filter or JPDAF (Bar-Shalom et al. 1980) is an extension of the Probabilistic Data Association Filter (PDAF) which handles the problem of associating an arbitrary number of measurements to an arbitrary number of targets. The literature on Condensation and multiple target tracking will be considered more fully in Chapter 4.

Once tracking has been carried out, it may be necessary to convert the tracked path in the image plane into a ground plane path for further processing. This is because the image plane is rarely parallel to the ground plane, and so some transformation of coordinates must be made. This is particularly important in a multi-camera system, as the ground plane is common across all cameras (Makris and Ellis 2002). Methods include recovering the ground plane by tracking

moving objects in the scene (Bose and Grimson 2003), and also from known lengths and angles of structures in the image (Liebowitz and Zisserman 1998).

2.2.4. Tracking as a method for surveillance of human activities

Monitoring and surveillance can be thought of as accomplishing the same task but for different reasons. Monitoring can be thought of as observing or estimating some internal properties of the targets, whereas surveillance is typically observing the target's potential effect on the environment. Monitoring exists in both invasive and non-invasive forms, the latter of these may involve some remote observation method, which in this case is visual tracking. For the purpose of this thesis, this non-invasive form of monitoring and surveillance can be thought of as interchangeable concepts.

A background on *human* surveillance will be presented first, as this is possibly the most studied incarnation of visual tracking of targets through a scene. Surveillance can be seen as two distinct and challenging phases: first, the tracking of the object through the scene, and second the description of the activities taking place. Monitoring breaks down in a similar fashion. Typically, image processing techniques like those described in Section 2.2.2 are combined with some tracking methods such as those described above to provide trajectories of the objects of interest. This is followed by some higher-level interpretation of the tracking results. Clearly, the first part of the surveillance problem requires accurate tracking techniques to be used as a foundation for all further processing.

The eventual aim for the surveillance of people is typically the automatic identification of certain activity types: typically suspicious or unusual behaviour. As humans, we find it easy to notice when someone is behaving in a fashion which is suspicious; noticing when someone is lurking in a car park, for example, or when someone seems to be following you. Progress is being made in automatically recognising similar behaviours from CCTV-style surveillance systems, but this is a challenging problem. Systems exist to generate models of pedestrian pathways in a scene and potentially identify suspicious or unusual

paths (Johnson and Hogg 1996), and other systems can label the behaviour associated with these trajectories (Makris and Ellis 2002). Surveillance work today characteristically consists of one or more static cameras overlooking a public space. Work developed at Leeds to track both pedestrians and vehicles (Remagnino et al. 1997) uses a single camera overlooking a car park. Using multiple cameras allows more ground to be monitored simultaneously (Grimson et al. 1998), as well as allowing cameras to be self-calibrated by tracking objects in overlapping images. More ambitious surveillance projects employ multiple overlapping sensors to try and track targets (Collins et al. 2000). Additionally, Collins et al. attempted to integrate several different *types* of cameras, from mounted CCTV cameras to thermal cameras, airborne cameras and very wide angle sensors. The system is essentially an amalgamation of different algorithms, each significant in their own right. How well they function when combined, however, is not clear. The system is tested by observing the quality of function in a real world situation, but no quantitative results are presented. The system appears successful to some degree, although the milestones achieved are presented as disparate sections rather than as a complete surveillance system – the difficulty with such systems is normally with the integration of techniques.

As extensions to just observing targets, further processing can lead to judgements about activities taking place in a scene. Behaviour detection systems can identify multiple agent interaction, for example people being dropped off from cars (Ivanov and Bobick 1999), or queuing for a bus (Grimson et al. 1998). Path detection systems can identify suspicious routing behaviour (Makris and Ellis 2002). A system for detecting abandoned packages in realistic situations has been developed by Beynon et al. (2003).

What this automated surveillance work demonstrates is that motion tracking through a scene can provide a wealth of information to higher-level processes, be they activity monitoring, behaviour labelling, common path detection, or calibration. Just as for humans, it is the visual observation of the motion of subjects that allows higher cognitive processes to determine whether the motion exhibited is normal or unusual; this is also true for automated systems. Despite

some successes, however, the development of a fully automatic, robust and accurate surveillance system is still held back by major technical challenges, both in the tracking and event analysis stages and in the setup of equipment (Dick and Brooks 2003). Progress continues to be made in this flourishing field, and the high demand for such a system ensures plenty of interest is maintained in the research community.

2.2.5. Animal monitoring

It has been seen that there exists a large body of research involved in the monitoring and surveillance of *human* activities. The main literature in the animal monitoring world shall now be discussed. The idea of looking for outlier behaviour in human activity has parallels in the animal world. Unusual behaviour amongst animals can suggest injury or disease, or other conditions such as oestrus (readiness to mate) or parturition (childbirth) (Frost et al. 1997). Within-animal motion differences can be detected, for example to tell a healthy cow from a lame cow by its gait (Magee and Boyle 2002). Magee and Boyle used two models of motion: a healthy model and a lame one, and the model that best represents the data is propagated, and can therefore be used to apply a label of lame or healthy to the animal in question.

A variety of animals have been the subject of research in the tracking and monitoring domain. Broiler chickens have been tracked (Sergeant et al. 1998; Bulpitt et al. 2000) with the aim of overcoming the subjective, invasive and tedious nature of human-observer experiments. Work over several years has used image analysis to automatically estimate the weight of pigs (Schofield and Marchant 1990; Marchant et al. 1999), and a system has been developed to track lab rodents (Branson et al. 2003; Westphal 2004). The rodent tracker, although only tested on a 30 second sequence, appears to work well, but the environment and lighting are quite tightly constrained. Work has been carried out building 3-D models of live pigs (Wu et al. 2004) with a view to using these models for further conformation analysis. 3-D information can convey information 2-D images cannot, such as allowing judgements to be made about the ‘squareness’ of certain muscles, which can be an indicator of lean muscle mass. All these systems

provide quantitative measurements of parameters (e.g. conformation and location) that are otherwise difficult to measure automatically. These systems are also completely non-invasive and so incur as little stress as possible on the animals being monitored. This is important both for the well-being of the animal and to prevent stress affecting the quality of the measurements.

The robotic sheepdog project (Vaughan et al. 1998; Vaughan et al. 2000) is an interesting example of the crossover between the monitoring of and interaction with a group of animals. A robotic 'sheepdog'(Vaughan 1999) was used to herd ducks to a defined goal, controlled by image analysis software (Sumpter 1999) monitoring the flock from an overhead camera.

One of the major commercial systems in tracking, monitoring and behaviour analysis is the EthoVision system (Noldus et al. 2001). Billed as a video tracking, movement analysis and behaviour recognition system, it is designed as an aid during behavioural experiments with subjects of the experimenter's choice. For example, it can be used to track and identify behaviours in insects (Noldus et al. 2002). Although the system has been used in many behavioural studies, it is more of a tool than a fully automated system. EthoVision can only automatically record a limited number of 'behaviours': whether an animal is moving and its movement parameters (such as location, path shape, proximity to others), and whether a rodent is 'rearing' (standing vertical) as decided by viewable surface area. The user of the system must enter any other behaviour observed manually via the keyboard or mouse. This is tedious, as the sequence must be watched by a manual operator to pick out other behaviours, and must be prone to operator error. Clearly this is not ideal for long sequences of video, but at the moment this is the state of the art for animal observation experiments. As for tracking, the system can track up to 16 unique (different coloured) targets in the scene, but the authors admit that the tracking of crowded targets is beyond the scope of EthoVision as it stands (Noldus et al. 2001; Noldus et al. 2002). Also, the quality of the tracking is not defined. The example studies listed in these papers suggest that EthoVision is a very useful experimental tool, but it requires a high degree of user interaction, be it physically colouring the animals or insects to distinguish them from each

other (Noldus et al. 2001) or having to manually key in behaviour types as someone watches the camera feed or video. A more automated system would be beneficial in many situations.

More insect monitoring work has been carried out on the tracking of ant colonies in an arena consisting of nests and food sources (Balch et al. 2001). This work is later extended by Khan et al. (Khan et al. 2003; Khan et al. 2004) to take into account social interactions to enable more robust tracking in the presence of similar interacting targets. Khan et al.'s work will be fully reviewed in Chapters 4 and 5.

The motivations behind the remote monitoring of animals are plentiful. The earlier the signs of disease or lameness or the presence of a predator are detected, the earlier the issue can be resolved, with the combined effect of both increasing the quality of welfare of the animals and ultimately saving the farmer money. For example, with the pig conformation work mentioned previously (Schofield and Marchant 1990; Marchant et al. 1999), an animal growing abnormally could be detected early, and examined by a skilled human to perhaps reveal and treat the cause of the abnormality, saving the animal's life if possible. This means less suffering for the animal and more healthy animals to sell for the stockman. Also, current observation experiments require many man-hours of manual observation and data entry. Automating this process frees researchers for other work, and also provides an objective reasoning about actions, whilst eliminating errors brought about by repetitive human observation. Where automated tracking and monitoring excels is the ability to extract objective, quantitative measurements from potentially huge amounts of visual data. Financially, it is generally cheap and easy to install an imaging system compared to the alternative sensing arrangements, especially when multiple measurements can be taken (for example, size, weight, location and conformation) from a single camera. Vision systems are typically more adaptable too, with the possibility of developing new software with the same hardware setup to measure different parameters in the future.

2.2.6. Summary

This research highlights some examples and possible uses and benefits of animal and human monitoring, but there are always constraints and limitations involved. Systems are usually specialised – either tracking routes or detecting unattended baggage, for example, but not both. Although attempts have been made to integrate different types of surveillance systems (Collins et al. 2000), a truly stand-alone, automated surveillance system is still to be achieved. The current state of the art does have practical uses though: one human observer can only watch so many screens before attention is lost and information is missed, and subjective human judgements can be made objective by automating the monitoring process. Also, automated labelling of behaviour can lead to fast retrieval of significant events (video mining), which is of clear use to post-analysis of recorded video; especially when there is a large amount of video from many cameras dotted around a city, for example. For animal monitoring, tracking can provide a strong foundation for the development of further processing techniques. Such animal monitoring systems can save the industry money and improve the welfare of the animals, both of which are significant goals. Increased production intensity on farms requires that processes be automated, as there are not enough skilled operatives to do the job in the time available. The monitoring and controlling of the production process directly affects the quality of the final product, giving financial rewards to initial investment into these stages, opening the way to the introduction of monitoring technology onto farms.

Chapter 3: Monitoring an animal by visual tracking: Automatic sensor positioning over the back of a feeding pig

3.1. Motivation

As described in the previous chapter, automated monitoring systems can allow for an increased number of measurements to be taken compared to manual methods. This in turn can lead to faster detection of abnormal conditions in animals and their environment, and to an improved product and hence increased financial reward for the stockman. Backfat thickness is a particularly important metric for pigs because it allows the leanness of the animal to be estimated. This in turn has a direct bearing on the price at slaughter, and therefore the profit for the stockman. Knowing the conformation or physical characteristics of the animal, such as that indicated by the thickness of backfat, allows the stockman to maintain careful control over the development of the stock. The feed can be tailored to individual animals' needs to maximise their leanness and health, or suitable animals can selectively be chosen for breeding stock. The only way to directly measure this backfat is to bring a sensor – optical or, more typically, ultrasound - into contact with the animal. This must be done at a specified location on the pig's back, to ensure that measurements are consistent across animals, as backfat depth varies with location. The manual measuring of backfat depth is a time consuming and skilled operation, and currently takes place much less frequently than is ideal. Developing an automated system is therefore a sensible solution, which would allow more frequent measurements and provide the stockman with more valuable

data than is currently possible. This data can then be used to optimise the pig's growth leading to increased quality of life for the animal and increased profits for the stockman.

3.2. Aim

This work will extend existing non-invasive pig monitoring work to create a system that is capable of locating a sensor over the back of a feeding pig. In particular, it is concerned with the accurate *location* of the sensor rather than actually taking a reading or interpreting a backfat measurement. Therefore, no actual sensor reading will take place. Instead, a laser pointer on the end of the robot arm will be used to measure how accurately a sensor would be positioned in the horizontal plane in a real world situation.

The hypothesis of this research is two-fold. First, that a sensor can be automatically positioned on the back of a feeding pig with accuracy equivalent to a skilled human operator. Second, that the automated system would allow readings to be taken more frequently that would be expected on a typical working farm.

3.3. Previous work

Complete systems for automatically monitoring animals (Frost et al. 1997) now provide the stockman with more data about the animals than has been possible before. This information can be used to manage the stock with greater efficiency and success, allowing for higher quality animal welfare and increased profits for the stockman. Pigs are a good subject for monitoring because they are farmed on a large scale, and the quality of the carcass has a dramatic effect on price. For example, there is a range of nearly 30 pence per kilo depending on weight and backfat thickness (Stotfold/MLC 2001). Previous work on using image analysis to monitor pigs includes using the pig's plan view area to estimate their weight (Marchant et al. 1999). This is a completely non-invasive method using a camera mounted above the feeder. Once manually calibrated at 75 days old, a pig's weight can be predicted to within 1kg at 125 days old, using its plan view area. This accuracy is comparable to that of the weighing machines used during the trials, but of course has the advantage of requiring little manual labour from the

farm's often limited workforce. Additionally, attempts have been made to measure conformation by examining 3D images of pigs (Wu et al. 2004; Tillett et al. 2004). Landmarks were placed on curvature features generated from 3D models of the animals. These 3D models were in turn generated from three stereo-pair images of the animal, from orthogonal directions. Such landmarks may in the future be used to estimate features such as the P2 point (see Section 3.3.1, below) and muscle specifications, though at infrequent intervals due to the large processing time required to build the 3D model.

The information that can be gained from visual data alone is limited. Certain measurements can only be obtained by placing a sensor directly into contact with the animal; for example, heart and respiration sensors, temperature sensors - although estimates can be made using infrared images (Best 1981) - and body composition sensors. A logical extension of a non-invasive system would be to make a system capable of taking invasive, contact measurements, guided by image analysis using the same camera configuration as for the non-invasive measures. Therefore the original non-invasive system could still be run with the same hardware set up. This is the reasoning behind this work. The system will take the form of a robot arm bringing a sensor into contact with a pig's back while it feeds, guided by image analysis software that tracks the pig from an overhead camera above a standard-specification feeder.

3.3.1. Body composition monitoring: Backfat and the P2 position

The price of an animal at slaughter is directly related to how lean the animal is. With an ideal weight animal, a difference in backfat thickness of 6mm can change the price of an animal at slaughter by 6 pence per kilogram (Stotfold/MLC 2001). This can be a change of about £4 per animal at slaughter for an average weight of about 70kg (Meat and Livestock Commission 2004). Clearly then, optimising this fat level is something every pig farmer should be concerned with.

This work will build an image analysis system to control a sensor placement robot and move it to the P2 point, over the back of a feeding pig. This point is an industry standard position for taking backfat readings on a pig using ultrasound

sensors. The P2 point is located at the last rib on the pigs back, 65mm from the dorsal mid-line (Frost et al. 2000; Youssao et al. 2002). Backfat measurements from this point enable an estimate of the leanness of an animal to be determined, and thus price at the time of market for the animal can be estimated, and its diet tailored to improve the leanness if necessary. An example of the manual operation required to take a measurement can be seen in Figure 3.1.

Figure 3.1 Manual ultrasound measurement of backfat thickness at the P2 position, taken at the same time as weighing the animal

Some ultrasound probes produce a 2D ultrasound image for the operator to interpret (ECM 2005) – these would be hard to analyse automatically, and would require further processing of the images to estimate backfat thickness. Other commercial probes are available that can quantify backfat using ultrasound when placed at the P2 position (Renco Corp. 2004). The placement system in this work is well suited to future adaptation to actually take backfat readings using this type of instrument. Automated backfat measurements are also possible from *carcasses* using machines such as the AutoFOM (Brøndum et al. 1998), which pulls a carcass across an array of ultrasound transducers and analyses the resulting 3D image to determine fat level. Such machines can grade carcasses at a rate of 1150

per hour, but of course this type of manipulation is not possible with live animals. More details on carcass inspection and backfat measuring techniques, as well as many statistics about animal weight and price, are summarised in the MLC Pig Yearbook, released annually (Meat and Livestock Commission 2004).

3.3.2. Robotics

The advantages of automating tasks by using robots can be seen in a wide array of fields. Robots are well suited to tasks which are repetitive (e.g. packing items), involve heavy manual work (e.g. automotive manufacturing industry) or in hazardous situations (e.g. inspection of nuclear power plants). In a sense, stockmen are sometimes faced with all three of these scenarios: having to work with many animals at frequent intervals, each of which is heavy and capable of causing injury or spreading disease, and also working in an environment which may be uncomfortable. Robotics have been put to use in agriculture typically to do a job that is time consuming to a skilled operator, for example cow milking (Frost et al. 1993b) or sheep shearing (Trevelyan 1989).

The environments in which an agricultural robot must work are often demanding. The type of power used to operate them must be chosen carefully. There are three main ways of providing power to a robot's actuators: hydraulic, pneumatic and electrical:

- Hydraulic: powered by the pumping of hydraulic fluid through valves and into the activators, hydraulic robots provide the greatest amount of power. They are also very accurate. On the downside, the hydraulic fluid is often toxic and so leaks can become a safety hazard, especially if operating in an environment with animals. Being powered by non-compressible fluid, the joints have no give and so are more likely to cause injury than other power types. This is also typically the most expensive option, being about twice as expensive as pneumatic or electric actuation (Frost et al. 2004).
- Pneumatic: similar to hydraulic, but with air instead of fluid being used to operate the actuators. The actuators typically have a 'spongy' feel to them because of the compressibility of air, so the robot can do less damage than if powered by fluid. Also, being supplied with compressed air, a leak is

not a safety issue. Accuracy issues include friction on the actuators causing 'sticky' joints, and oscillation caused by overcorrection.

- Electric: a compressed air or hydraulic fluid supply is not required. However, electrical components in some operating environments may present a risk of electric shock. This is especially true in agriculture where the environment may be damp and many animals are present.

3.3.3. Image analysis for tracking pigs

Pigs and indeed any animals present a challenge to automated tracking systems. Their shape is deformable, and their appearance liable to alter due to lighting changes and the build up of dirt on the animal. Previous work exists for tracking animals in certain situations, including cows (Magee and Boyle 1999; Tsutsumi and Kita 2002), duck flocks (Sumpter et al. 1997), tracking mice in cages (Branson et al. 2003; Westphal 2004) and tracking poultry (Sergeant et al. 1998). Some work already exists for tracking live pigs from an overhead camera. Complete contours of pigs have been tracked from an overhead camera (Marchant et al. 1999), and point distribution models (Cootes et al. 1992) have been used to track animal movements (Marchant and Onyango 1995; Tillett et al. 1997). Particularly of note is the Snake algorithm (Kass et al. 1988) which has been used successfully to locate the boundaries of pig outlines from an overhead camera (Marchant and Schofield 1992). Snakes are a type of deformable model (also called Active Contour Models). These models are very versatile, and have been used to good effect in a variety of fields, including animal tracking, medical image analysis (McInerney and Terzopoulos 1996), hand and gesture tracking (Heap 1995; Blake and Isard 1998) etc. Snakes will be described in detail in section 3.5.2.

3.4. System specification

3.4.1. Accuracy

The effectiveness of the sensor placement will be measured by comparison to pre-determined accuracy limits. Tillett et al. (Tillett et al. 2002) suggest limits of 25mm longitudinally and 10mm laterally in order to measure a backfat depth to within 5% of the actual value. This actual value is at a local minimum in backfat depth and so repeated measures can be taken and the minimum of these used as an estimate of back fat depth. The working limit of a human operator has been estimated to be within about 20mm from the P2 position (Frost et al. 2000). Therefore, the robot can be considered an improvement on human sensor placement if it achieves more readings per day of or above equivalent human accuracy i.e. the laser spot is within 20mm of the P2 position. It was felt that comparing to human accuracy would allow a fair comparison of the new robotic method to existing manual methods. In a real-world system, the actual P2 backfat depth could then be identified from multiple readings by identifying the minimum value.

3.4.2. Overview

Figure 3.2 provides an overview of the system. The physical arrangement of the system components was based on a similar arrangement that had been previously used to estimate the weight of pigs (Marchant and Schofield 1992; Marchant et al. 1999). More streamlined software was written for locating the P2 point, which is real-time critical, and so the weight estimation software was left unmodified by this work. However, with the current hardware configuration the original software could be run at the same time, thus providing weight and backfat estimates from one hardware installation.

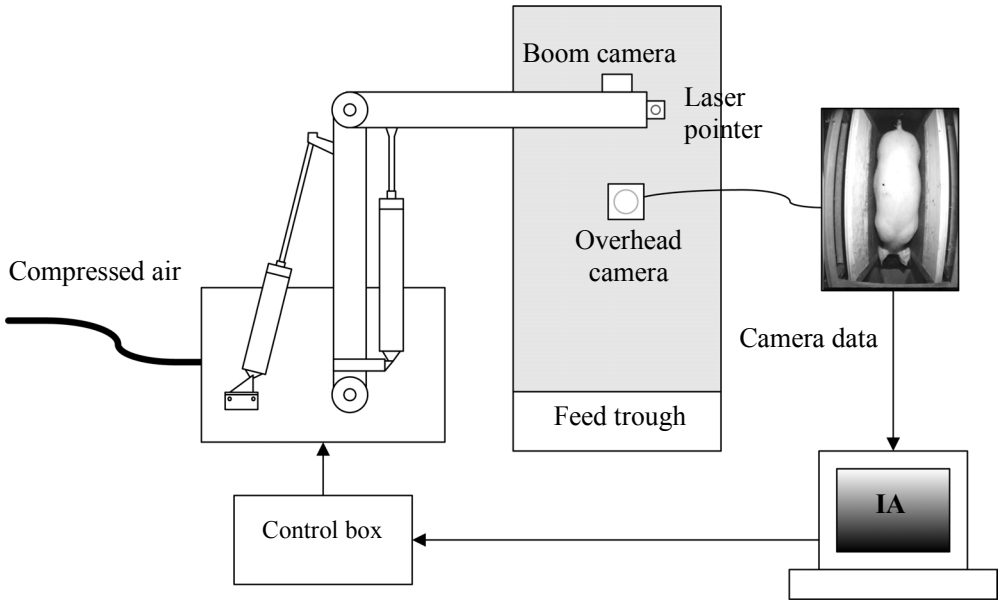


Figure 3.2 Diagram illustrating the components and connectivity of the system

3.5. Image Analysis and Locating the P2 point

3.5.1. Aim

The aim of the image analysis component is to determine the P2 location from features in images of the animal taken from an overhead camera, and communicate these coordinates to the robot control system when the pig is considered sufficiently stationary.

3.5.2. Image processing steps

The P2 position itself cannot be directly identified, as it has no visible features. When manually located, it is normally found by an expert palpating the back of the animal. It will be located in images by modelling the position of the P2 point from the location of certain physical features of the pig which *are* observable. To model the P2 position, previous work (Frost et al. 2000; Tillett et al. 2002) has shown that tracking certain visible feature points on the boundary of an animal can produce a sufficiently accurate estimate of P2. These curvature features that will be required for this work are the 'kink points' on the boundary of the animal. These occur on the boundary between the rump and the abdomen, and between the abdomen and the shoulder as shown in Figure 3.3.

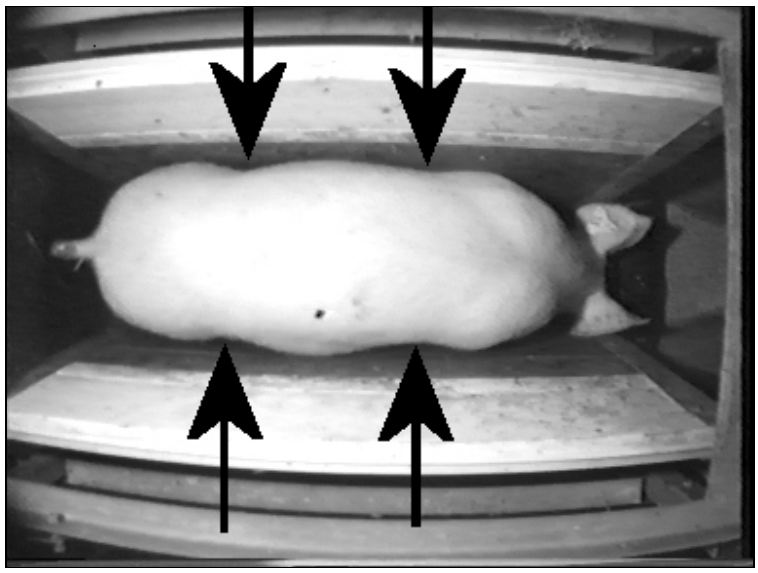


Figure 3.3 Arrows indicate the 'kink points' on the pig's boundary.

More details on these features and the P2 model (section 3.5.4) will be presented shortly, but from an image processing point of view it is only necessary to know that these kinks are features that can be detected visually, and their location will be used as input to a model of P2 position. The steps required to locate these kink points will now be presented.

The first step in any system such as this is to identify when there is any action taking place in the frame. In this scenario, most of the time the feeder is likely to be empty, and identifying when an animal enters the feeder is necessary. In the image frame, the presence of a pig can be determined by looking for an increase in intensity at a spot where the pig is definitely going to be when feeding. A suitable location for this 'hot spot' (Marchant et al. 1999) is at the area the shoulder of the pig would occupy if it had its head in the feeding trough. Once a pig is present, it is possible to begin locating the P2 point.

A pig in the feeder presents a slightly curved surface to the camera. As there is a light directly under the camera, this surface becomes increasingly illuminated the closer to the camera it gets. Knowing this fact, thresholding is used to locate the brightest section of this curved surface, and because of the known geometry of the animal we can use this bright section as an estimate of a central region in the rump half of the pig's boundary. Such a threshold leads to an output as indicated in Figure 3.4:

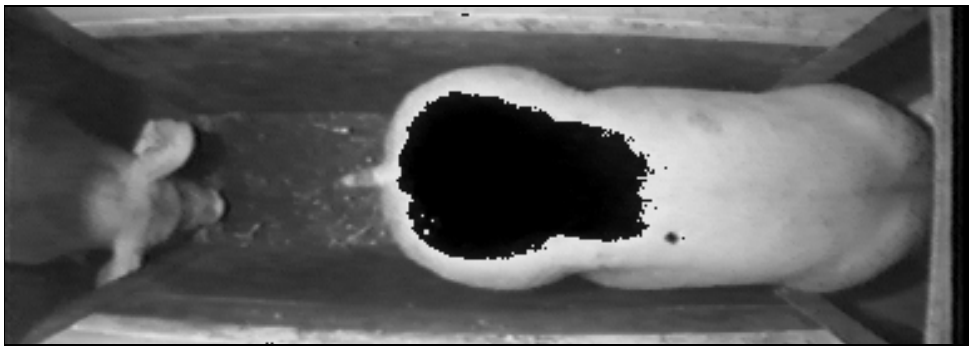


Figure 3.4 Example of area selected (in black) after thresholding with a limit of 230.

Once the highlight has been found, it is straightforward to detect the extreme bounds of this thresholded region in both the x- and y-directions, and these anchor

points within the boundary can be used to initialise the further image analysis procedures.

The value used to threshold this region could either be set manually, or it could be determined as the animal enters the feeder. To determine it on the fly, an estimate of the average grey-level of an animal is required. One method of estimating such an average is to use background subtraction to isolate the foreground pig pixels. A background image was taken of an empty feeder, and then this was subtracted from a new image when a pig was detected in the feeder, giving a result such as Figure 3.5:

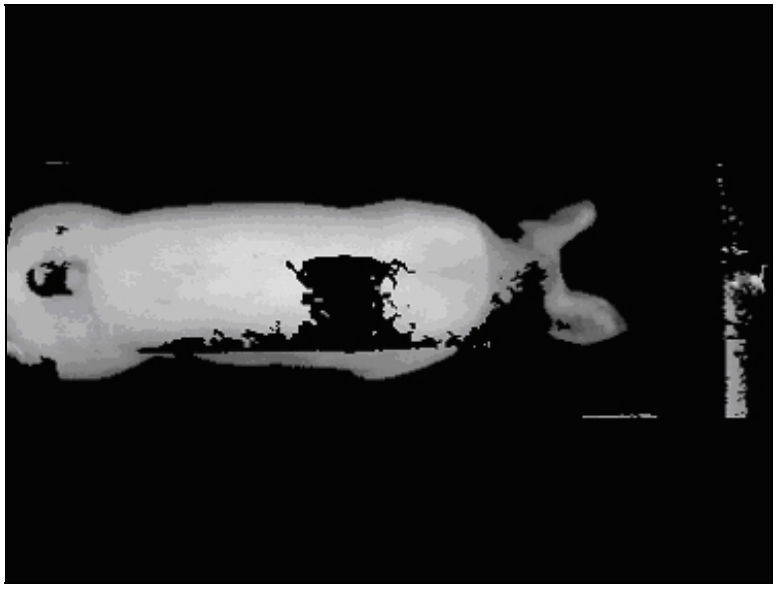


Figure 3.5 Background subtraction-generated image. Note the missing foreground pixels

As can be seen in Figure 3.5, this process by no means gives a reliable outline of an animal, and it does pick up some background clutter, but despite this *most* of the pixels do originate from the animal. Working out the average value of these pixels gives an estimate of the average intensity value of the pig, and from this a suitable threshold to select the rear region can be calculated. However, in practice, as illumination remained constant and the animals could be treated as being uniform in colour and reflectance, a constant threshold was found to work well. If the algorithm were to be implemented in a real-world system, certainly one where the ambient illumination is likely to change significantly, then such an

active, real-time approach would be beneficial as it would allow for long term illumination changes and varying reflectivity of the pig's surface.

For the P2 model, only the positions of the kink points are required as parameters. Therefore, tracking the whole boundary of the animal was not thought to be necessary. In the interests of performance, complete-boundary tracking algorithms as used for surface area calculation in previous work (Marchant et al. 1999) were not implemented. Instead, it was only necessary to locate particular sections of the boundary to identify the required features in each frame. Previous work (Frost et al. 2000; Tillett et al. 2002) has used six feature points on the boundary. Four of these features were the 'kink' points on the boundary that occur in front of the hind legs and behind the forelegs. The other two were at the base of the tail and an estimated point on the neck boundary. It was hypothesised that processing speed could be decreased by only tracking the minimum necessary parts of the animal's boundary (i.e. those around the feature points). The sixth point at the neck was considered inaccurate as it occurred on a hypothetical boundary and so was dropped. The point at the tail was also dropped as it was not considered to provide much more information than the kink features can give, and also its perceived position can change depending on the relative positions of the pig and the camera. This leaves a set of four features, the boundary kink points, which bound the P2 position (Figure 3.6).

.

Figure 3.6 The four boundary kink detectors, and the predicted P2 point resting over a pre-marked P2 point for this test image.

The accuracy of this new four-point model was compared with the old six-point model and found to be acceptable – refer to 3.5.4 for more details. Therefore, only the sections of boundary that contained these four points were located. In order to

fit models to the side of the pig that trace the boundary around the kink points, a variation of a Snake (Kass et al. 1988) was initially used. Snakes are deformable contours (a form of Active Contour Model) that have underlying geometrical or physical constraints, typically within a system of Lagrangian mechanics. The snake has a measure of energy depending on its position and shape; this energy consists of an internal component, related to the geometry of the contour, and an external component that is related to the image, equation (1). The snake will seek to minimize the energy of the system by deforming appropriately. The internal energy consists of a parameter controlling the curvature of the snake, and a stretching energy. These parameters can be altered to suit the features of the intended target. The external energy function is based upon appropriate features of the image, such as edges for locating boundaries. The final shape of the snake is determined by a minimal energy state balancing all the functions.

$$E_{Snake} = E_{Internal} + E_{External} \text{ , where } E_{Internal} = (bending+stretching) \qquad (1)$$

Snakes have been used successfully to locate the boundaries of a number of 'rounded' targets in images, including cell membranes (McInerney and Terzopoulos 1996), fruit (Kass et al. 1988), heart ventricles (Cohen 1991) and complete pig outlines (Marchant and Schofield 1992). Extensions to snakes have allowed the localisation of more complex objects such as hands (Cootes and Taylor 1992), and addressed some of the problems associated with Snakes, such as looping when tracking irregular objects (Ji and Yan 2002). For the specific purpose of locating the contour of a pig around the kink points, some adaptations to the original Snake active contour formulation are needed here. First, the original Snake is a closed loop, fitting over image features like an elastic band. To detect the contour segments of a pig, only a model of a line is needed for each kink. As can be seen in Figure 3.14 on page 53, these linear Snakes only detect a small portion of the whole contour. The second change is that the snakes do not require a bending energy. A consequence of having a limit to the amount of curvature allowed is that the snake tends towards a straight line (Perrin and Smith 2001). Knowing that the edge we are interested in detecting will contain a point of sharp curvature, having a model that prefers straight lines is not desirable.

Previous work using snakes to detect pig contours (Marchant and Schofield 1992) also disregarded the bending component for presumably similar reasons. The stretching component is another function of snakes that is largely redundant in this application, particularly as lines are used rather than closed loops. Stretching tends to contract the line, and as there is no corresponding force to 'pull' the line out again. It also has the tendency to 'iron out' kinks in the line, which is very undesirable for this work. Instead, a string of points which move as a rigid group longitudinally (constrained by rump location, explained below) but that can move independently laterally is used. The traditional internal energy of the snake has then been effectively removed, as both bending and stretching can produce undesirable effects for this situation. This new snake-like structure has evolved into a localised linear contour detector, which is exactly the role required of it for this work. This contour detector is driven laterally outwards from the centre of the pig and positioned longitudinally according to the position of the rump: these properties could be considered as variants on the traditional internal energy components of a snake.

The placement of these four contour detectors was initially dependant on the location of the rump-end of the thresholded bright area on the rear of the animal. However, it was desirable to have a more accurate location of the rump, as it was found that the left-most extreme of the thresholded region was subject to a lot of variablility; typically as the pig moves away from under the spotlight, and especially later on when the robot was present and forming shadows on the rear of the animal. Therefore, an extra contour detector is used, and is initialised over the rump. This is accomplished using the information about the thresholded highlight to determine the approximate longitudinal centre line of the pig, and then a snake is moved from the left edge of the image along this line, until it finds the rump contour and 'adheres' to it, as in Figure 3.7.

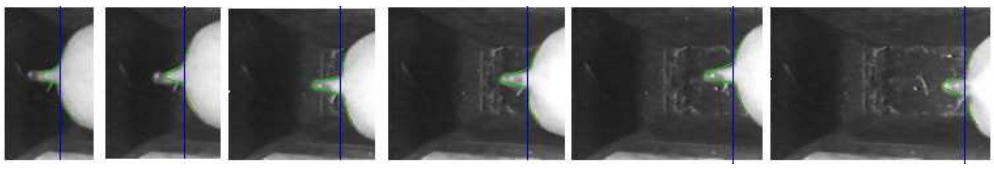


Figure 3.7 Rump snake contour detector finds the rump contour in a variety of images.

Tracking the rump with a snake has an additional benefit. When the robot moves over an animal to place a sensor, the robot arm obscures one or more of the kink points, as in Figure 3.8:

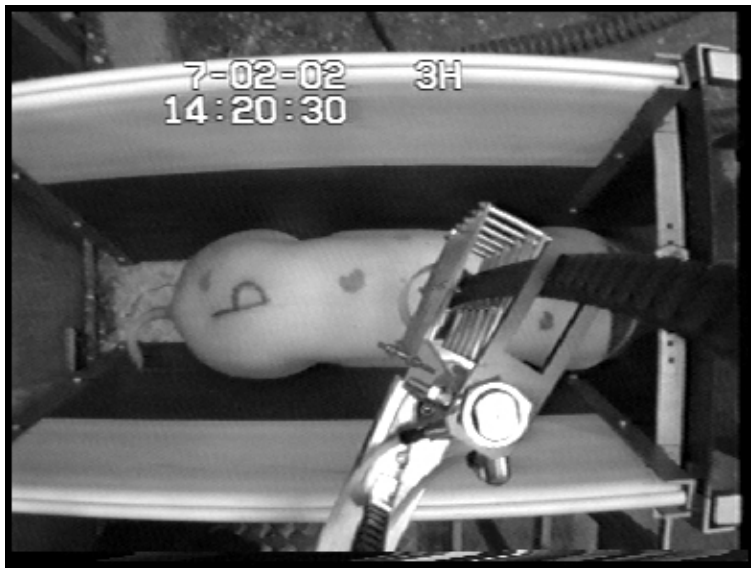


Figure 3.8 Example of how the robot arm obscures some kink points, but leaves the rump clearly visible. The rear kink features are often obscured as the robot moves.

Because of this effect, the P2 point cannot be predicted when the robot has been activated because some of the model parameters cannot be supplied, which is why the robot cannot operate in a 'constant tracking' mode. However, by tracking the rump alone when the robot has been activated, the system can make an estimate of how much the pig is moving longitudinally. Longitudinal movement forms the greatest component of the overall motion as the animal is constrained laterally by the walls of the feeder. Therefore, when some 'movement threshold' is exceeded longitudinally by the rump, the robot can be automatically reset.

Once the rear end of the pig's contour has been detected, an estimate of the rear end location is calculated by taking an average of the x-coordinates on this 'rump detector' (the vertical line in Figure 3.7). This has the advantage of largely discounting the extrusion caused by the tail, the length and position of which can vary. This position can then be used to place the four 'kink detecting' snakes. These kink-detectors are placed along the approximate centre line of the pig, longitudinally positioned using the rump snake as a reference point. Once the

kink detectors are positioned longitudinally, they search towards the outside of the pig's contour looking for the edge. The external energy for the detectors is based on an edge detector filter response. The main filter used is a Canny edge detector, and a typical repose is given in Figure 3.9:

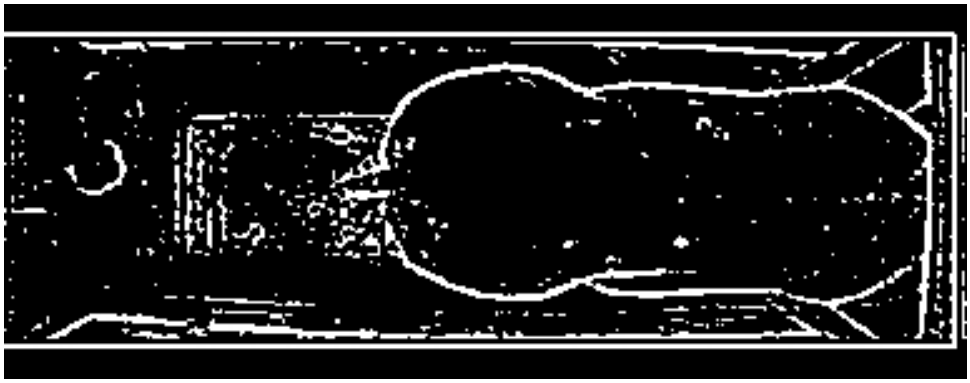


Figure 3.9 Edge detector response for a typical frame

It was initially intended that the snakes would track the edges by maintaining their shape from one frame to the next, moving longitudinally depending on rump movement and then updating their shape by searching out the boundaries in the local neighbourhood. Each point of the snake is effectively constrained in the x-direction by the position at which the rump is detected. However, each point can move independently in the y-direction. Each point searches a small number of pixels in the y-direction looking for an edge. Thus each point effectively becomes a one-dimensional edge detector. A similar approach of searching along curve normals for edges has been used when fitting curves to objects (Blake and Isard 1998).

This rudimentary tracking approach worked well in general situations, but in some cases where particular background clutter existed, the detectors would become caught on an edge caused by clutter, and as they maintained their shape from frame to frame, they would be unable to reset their position on the correct boundary. However, as the developed partial-boundary, simple-snake approach runs so fast, it is feasible to effectively re-initialise the detectors every frame. This was found to work very well in a real-world situation where reliable tracking was required.

Once the kink detectors are positioned on the boundary, it is possible to extract the actual 'kink point' from the line in a variety of ways. One way is to examine the curvature of the line and see where the 'knot' is, or where the curvature inverts. With smooth kinks containing noise, however, this approach was found to be unstable. Another method that was tested was that of fitting circles onto the line and examining where they crossed:

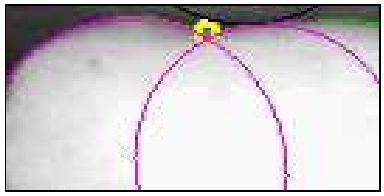


Figure 3.10 Example of using circles to successfully identify the kink point.

Such an approach was found to be slow, and to prevent the circles fitting to noise an estimate of the initial location of the kink point was required (typically in the middle of the line), which was not always a valid assumption. Both of these techniques would fail if the kink detector were slightly misplaced to the side of a kink, as they would be presented with a line with no kink point, as in Figure 3.11:

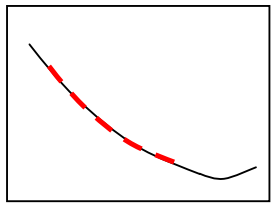


Figure 3.11 Example of a slightly misplaced kink detector. Using the circle-fitting or knot-finding methods would fail to find the closest point to the true kink point.

One method that would be much more successful here, and one which is also very fast, is to treat the line as a graph and find the minimum point. This can be practically found by looking for the minimum y-coordinate in the detector (or maximum, depending on the side of the animal). This method was found to work very well and so was employed in this work. Once these points have been found, they can be used in the four-point P2 model (considered in section 3.5.4) to predict the P2 position, and from there direct the robot.

One possible extension to improve the location of the kink positions would be to incorporate any available *symmetry* information into their placement estimates. When observing the animals, there does appear on occasion to be an element of symmetry between the motion of two sides of the pigs. This is to be expected to an extent as obviously the two sides of the animal are connected. However, when a pig twists in the feeder, or during a step (which introduces a characteristic wobble into an animals motion), the symmetrical link between the lateral sets of kink points appears much weaker. Therefore for this work it was not felt that adding a symmetrical component to the location algorithms would improve accuracy. However, if in later work there was found to be a symmetric relationship between kink points under certain conditions, then this relationship could potentially be used to improve the accuracy of placement or estimate the locations of ambiguous kink points when these conditions hold.

3.5.3. Detecting when the animal is in a motionless state

It was desirable to instruct the robot to move when the pig was in a motionless state. Previous motion graphs (Frost et al. 2000) and general experience with the animals has indicated that once a pig has started eating, it will remain eating, and therefore remain stationary, for a while. A lack of motion from the animal is a good indicator that it has begun eating from the trough. Identifying this state allows the robot to be activated and take a measurement at a time when the animal is most likely to remain still. To do this, the motion of the predicted P2 point was recorded over a sliding window of about the previous four seconds. If the variance of the motion within this window was below $40mm^2$, the robot was activated using the most current P2 estimate. The values for the time-length of the window and the distance of the threshold were determined by practical experimentation, larger windows and lower thresholds giving rise to longer periods of suitably low motion being necessary to activate the robot. Figure 3.12 shows how using a sliding window of variance calculated over approximately the four seconds can be used as an estimate of when the animal is in a motionless state:

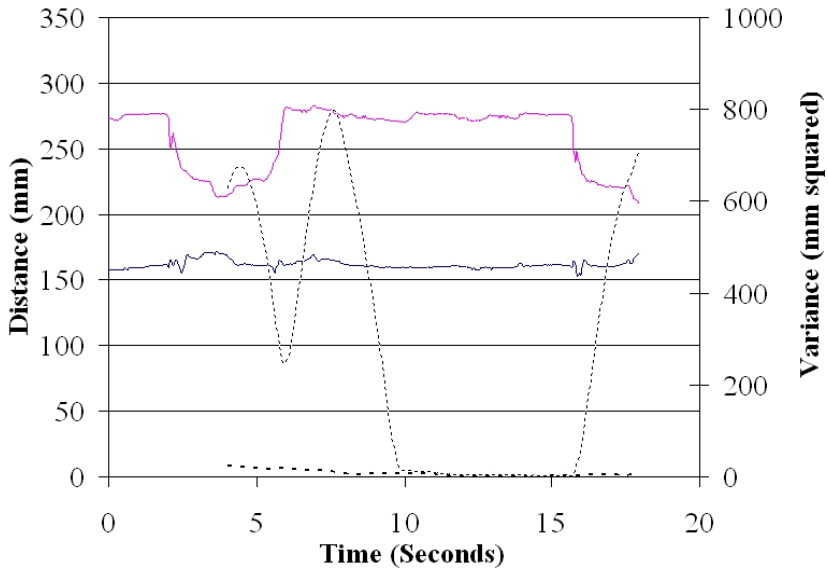


Figure 3.12 Graph showing the value of a variance window of P2 point movement calculated over approximately the last four seconds, and the corresponding pig movement values over time. The solid lines represent pig movement data (the blue line is lateral movement across the feeder and the pink link is longitudinal movement along the feeder) from a 20 second sequence of a pig in a feeder. Dotted lines are the variance (thick is lateral and thin is longitudinal).

It can be seen from Figure 3.12 that the animal is considered motionless (i.e. variance is less than 40mm^2) between approximately 10 and 15 seconds. The robot would be activated at 10 seconds, allowing 5 seconds for the robot to position itself, and then take a reading. In this example, it can be seen that variance of P2 movement over a four second window is a good indicator of the amount of motion the animal produces.

3.5.4. Model of P2 position

As the P2 point is determined by the internal structure of the animal and is not marked by visible features, it must be located by reference to other physical points on the pig. A manual operator would find the point by a combination of palpation and prior knowledge. While we cannot palpate, we can use a model of its location to determine where it is in relation to other parts of the pig's anatomy. Previous work has examined the feasibility of using features on the boundary of the pigs plan view to predict arbitrary points on the pigs back (Frost et al. 2000). It was found that using six feature points on the boundary of the animal, nine spot

positions distributed across the back of the pig could be located. This original model was based on six feature points: the four kink points and additionally a neck point and a rump point (Frost et al. 2000; Tillett et al. 2002). The neck point is artificial, as it exists on an imaginary boundary; it does not correspond well with actual visual features. The neck point is defined as where the major axis of the boundary area intercepts the boundary at the head end, and is thus a function of the boundary. The tail point is where the centre of the tail intercepts the periphery, and is sensitive to the viewing angle of the pig and the location of the tail. For these reasons, the model was tested without these two extra, less-reliable features; the coefficients were recalculated from just the four kink points on the 7549 legacy training images that were used for the six-point model. This new model was compared to the six-point model by computing RMS errors compared to a human marked groundtruth for a 450 frame sequence of pig motion in a feeder. The new model was found to be more accurate laterally, though only by 0.2 pixels (~0.4mm), and was approximately 1.3 pixels (~2.6mm) less accurate longitudinally. It is suggested that the lateral location of the neck and tail features can vary independently to the motion of the pig, depending upon orientation, and so provides a slightly less accurate lateral position for P2 than if these features are excluded from the model. However, the longitudinal positions of these extra points is less ambiguous and so does provide salient extra information to the model.

For the purpose of the sensor placement system, using the four-point model has several advantages. First, the whole boundary need not be tracked. To locate the tail and neck points, tracking nearly the whole boundary of the pig is necessary, but to use only the four kink features, only these sections of the boundary need be located (see 3.5.2 for details). This will save processing time, and work around the problem of the front area of the neck becoming obscured in the feeder, as occasionally happens. Second, the longitudinal reduction in accuracy for using only four features is acceptable, especially considering there is actually an improvement in accuracy laterally.

The P2 position was modelled from the located kink features found from the image processing stage described in 3.5.2 above. These four features were found in 7,549 frames from ground truth data used for previous work at Silsoe Research Institute (Tillett et al. 2002), together with a manually located P2 position located in each frame (as marked on these training pigs before image capture). The model is a linear regression of the spot coordinates on the kink coordinates.
The model used is of the form:

$$s_x = a + \sum_{i=1}^{4} (b_i k_{ix} + c_i k_{iy}) \tag{2}$$

$$s_y = d + \sum_{i=1}^{4} (e_i k_{ix} + f_i k_{iy}) \tag{3}$$

where (s_x,s_y) is the predicted coordinate of the P2 point, (k_{ix}, k_{iy}) are the coordinates of kink point i, and a, b_i, c_i, d, e_i, f_i are the model weights.

This model is based on data used in building the six point version, applied successfully in previous work (Frost et al. 2000; Tillett et al. 2002), modified for use with four feature points. As was explained earlier, the rump of the animal *was* located by a detector after all, and so theoretically a 5-point model could be used, incorporating the location of the tail. However, it was found that the rear contour detector is occasionally susceptible to tracking clutter on the floor of the feeder. While this does not greatly impact an estimation of the rump location, it would hamper any effort to determine the base of the tail. Combined with the orientation-dependant view of the tail, the rump detector was not used to detect the fifth feature point. Therefore for this work a 4-point model was implemented.

The a and d offsets in equations (2) and (3) above are allowed to vary from animal to animal, as previous work (Tillett et al. 2002) has shown this to be the most accurate model. This will be accomplished by setting these parameters manually for each animal, although in the future this need only be done once and then, by tagging the animals electronically, the parameter values could be recalled automatically when the animal enters the feeder.

3.5.5. User interface

All the image processing is hidden from the user behind a graphical user interface, extended from a system called CamCap developed at the University of Nottingham[1].

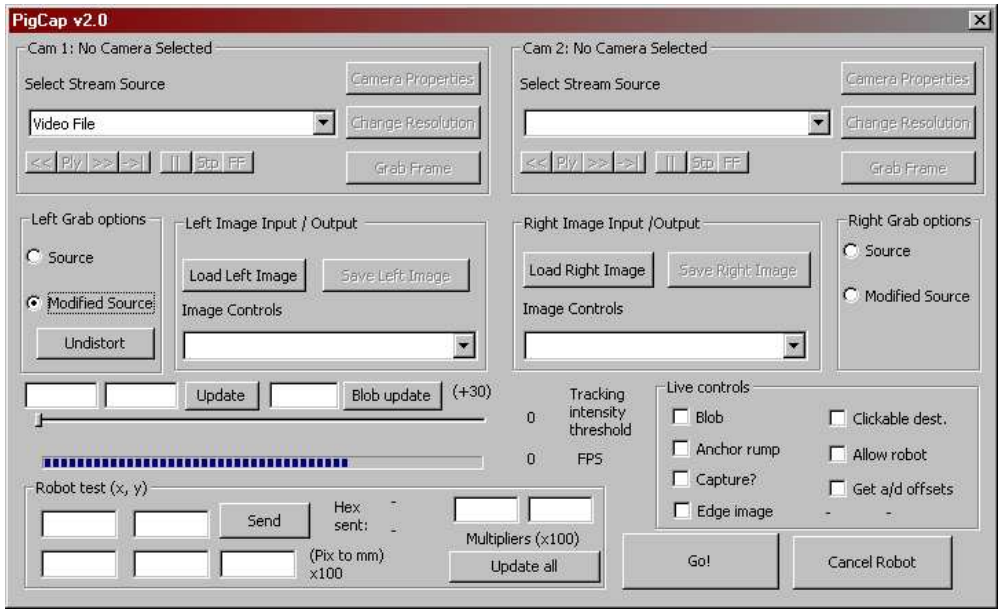


Figure 3.13 Graphical user interface from the image analysis software. The live video feed pops up in another window.

The CamCap program employs Microsoft DirectShow technology to handle image display, and Intel's Open CV library (Intel 2005) for some of the low level image processing algorithms.

[1] CamCap video processing environment by Jonathan Green, Andrew French. Other modules by other authors. See (Green and French, 2006).

3.5.6. Example of image processing results

After the image processing stage and having located the P2 position using the 4-point model, a typical frame now looks like:

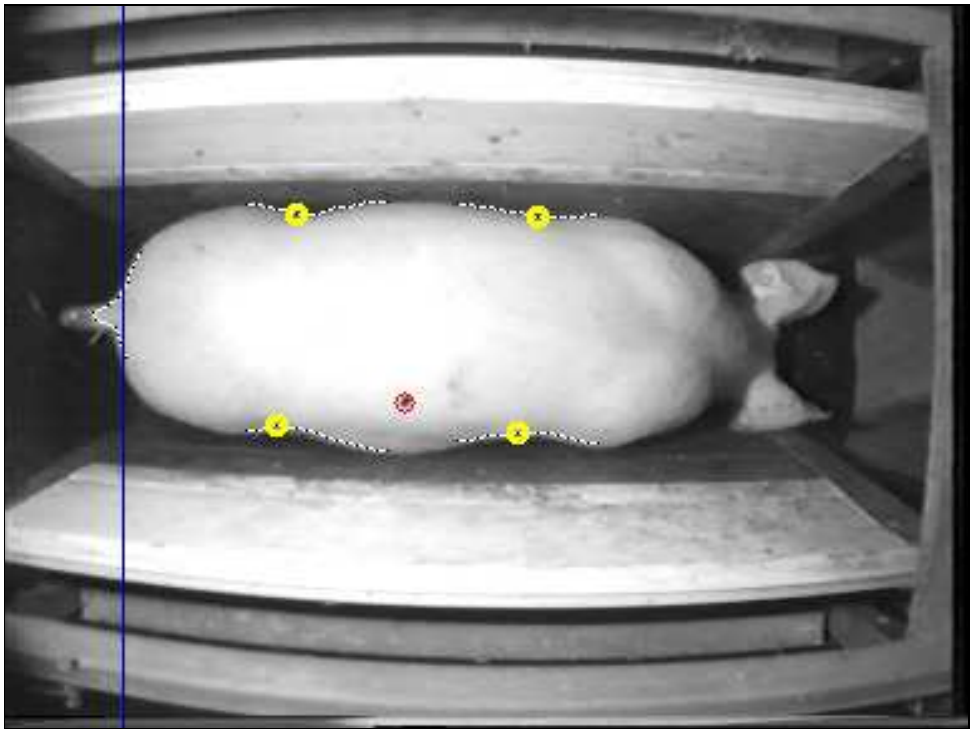


Figure 3.14 The four kink points located along the length of the linear snakes. The P2 point is marked by a circle on the back of the pig: note how it correctly falls over the actual P2 position, as indicated by the black tape on the back of the animal.

The final system processes the frames at about 10 frames per second on a Pentium III 500 MHz machine. Optimized for a modern machine, this could be expected to run at a full 25 frames per second.

3.5.7. Image distortion correction

The images from the overhead camera suffer from two main types of distortion. The first, radial distortion, is due to the wide-angle lens that is used, and secondly the more minor decentering distortions (Conrady 1919); see Figure 3.15 on page 56. Therefore, it was necessary to determine whether these distortions needed to be removed. To estimate the error involved in predicting the P2 using a distorted image, the difference between the predicted position on both a distorted original image and an undistorted image was calculated for three test images. Two of

these images represent extreme conditions, where the pig is at the far left and far right of the image (where radial distortion is the most prolific) and one represented a typical situation where sensor placement would be likely to occur.

The images were undistorted using a blind distortion removal technique (Swaminathan and Nayar 1998). 'Blind' means that the parameters of the internal camera optics need not be known, and no specific calibration target is required. More traditional calibration methods require a calibration target, such as Tsai's popular work (Tsai 1986). Taking into account the environment in which this system must operate it was decided that manual calibration methods should be avoided if possible, therefore existing geometry in the image should be used to guide calibration. Known straight lines in the real world appear as curves in a radially-distorted image. For example, the feeding rig walls are straight in the real world but are curved in the images in Figure 3.15 on page 56 (left hand column). Swaminathan and Nayar propose a method whereby such a curve is warped back to a straight line by estimating the parameters of the distortion. Other distortion removal techniques were considered (Farid and Popescu 2001; Devernay and Faugeras 2001), but given the presence of straight lines in the image and the apparent success and simplicity of Swaminathan and Nayar's method, this technique was chosen. To test the effects of distortion, the four kink points were marked by hand, and their coordinates used to calculate P2 as per the model. The robot is sent an integer and so the values in Table 3.1 are as the robot would receive them.

	Distorted P2 coordinates		Undistorted P2 coordinates		Error	
Pig	**x**	**y**	**x**	**y**	**x**	**y**
1	114	169	111	170	3	1
2	286	162	285	162	1	0
3	162	160	161	160	1	0

Table 3.1 Estimated coordinates and errors of predicted P2 position in pixels, calculated for both distorted and undistorted images. One pixel ≈ 2mm. Pigs 1 and 2 represent extreme cases of distortion (where the animal is towards the edge of the image) and pig 3 represents a typical situation.

It can be seen from Table 3.1 that very little error is introduced as a result of radial distortion. The maximal error encountered was about 6.3mm in the Pig 1 image, where the animal is entering the feeder. The images of pigs in the expected sensing position (Pigs 2 and 3) produce only 2mm error. For this reason, it was decided that it would not be necessary to remove the distortion from the images before processing. Decreasing the time spent processing the frames was considered more important as this gives less time-lag in which the animal can move before the robot is activated. Additionally, the model presented in 3.5.4 was historically learnt from distorted images.

It should be noted at this point that only *radial* distortion was removed, as this was by far the main component of the distortion. Other smaller distortion effects may still be present.. However, it can be seen that most of the distortion is removed when radial effects are accounted for. Therefore, radial is the largest component of distortion, and even so produces only small errors in the final P2 predicted position. Other small distortion components need not be of concern for the accuracy required of this work.

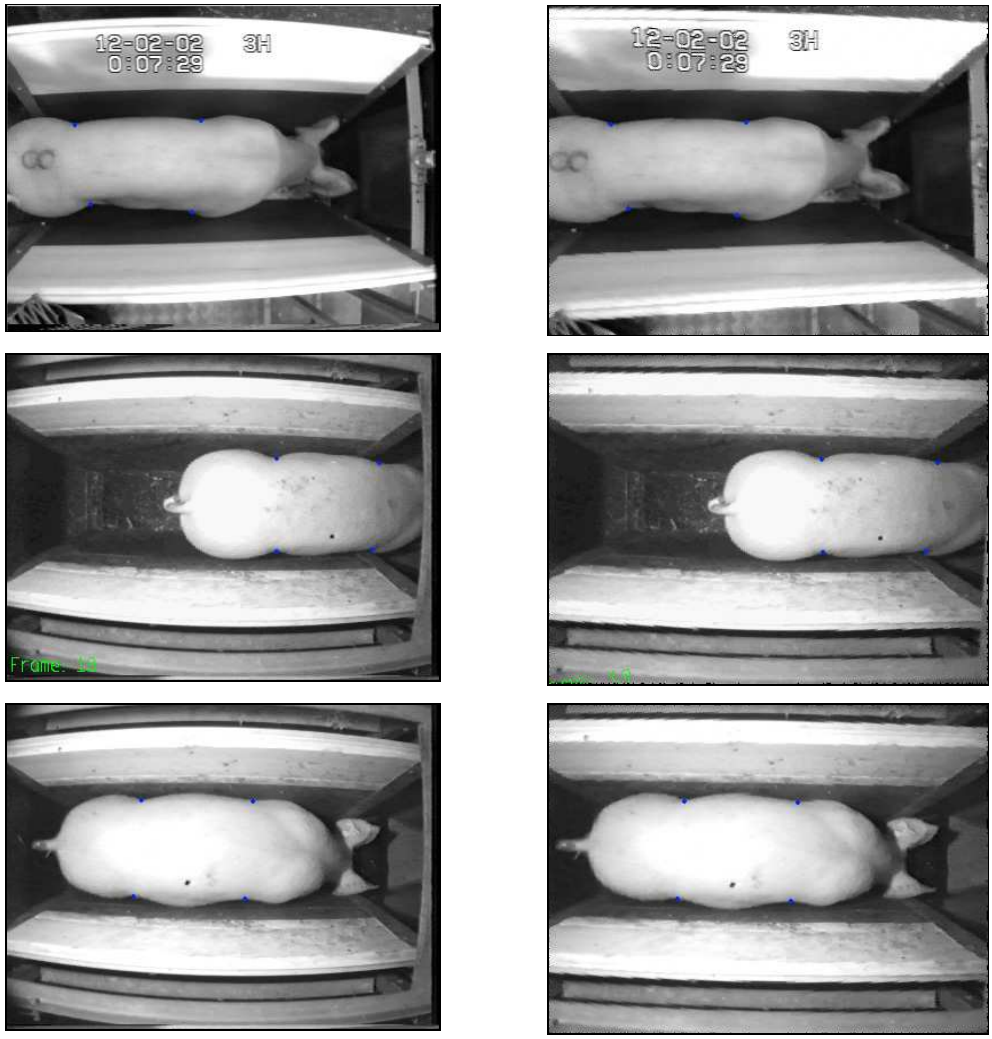


Figure 3.15. Distorted images (left) and with radial distortion removed (right)

Top: Pig 1. Middle: Pig 2. Bottom: Pig 3.

3.6. Robotics: specification and calibration

3.6.1. Specification

The choice of power for the robot is constrained mainly by the operating environment. Electricity was not an option in such an environment where powerful animals and damp conditions will be present. Hydraulic robots provide speed and accuracy, but the possibility of a leak of toxic fluid over an enclosure of animals whilst unattended is too great, plus the cost is prohibitive and there is a greater chance of impact injury. Compressed air offers a non-toxic system that has the advantage of some give if animals (or people) get in the way of a moving arm, or if it goes out of control, or a component becomes damaged. The automatic teatcup attaching robot (Frost et al. 1993b) employed pneumatics successfully; all four teatcups were successfully attached on 68% of trials (Frost et al. 1993a). For these reasons, a pneumatic robot was a sensible choice for this work. The compressibility of the air provides a cushion should the arm strike an operator or animal and should a leak occur, compressed air is harmless. Pneumatic robots do suffer from stiction (or 'static friction', an effect of friction which requires increased force to start moving two stationary surfaces in contact with each other.) in their actuators, and non-linearities due to the compressible nature of air, but offline tests (Frost et al. 2004) suggest the robot is capable of achieving the required accuracy.

The robot is theoretically capable of continuous tracking of a target position. However, for this work, a non-tracking method was employed, whereby the robot is sent to one position, and it remains there until reset. There are a number of reasons behind this decision. First, the image analysis is unable to continually predict P2 position when the robot has activated because of the obscuration of one or more kink points that typically takes place. Second, the robot was found to be insensitive to movements that were short (less than 2mm) or high frequency (above 2Hz) (Frost et al. 2004). Previous work estimated that non-tracking placement would still generate sufficient readings per day (Frost et al. 2000).

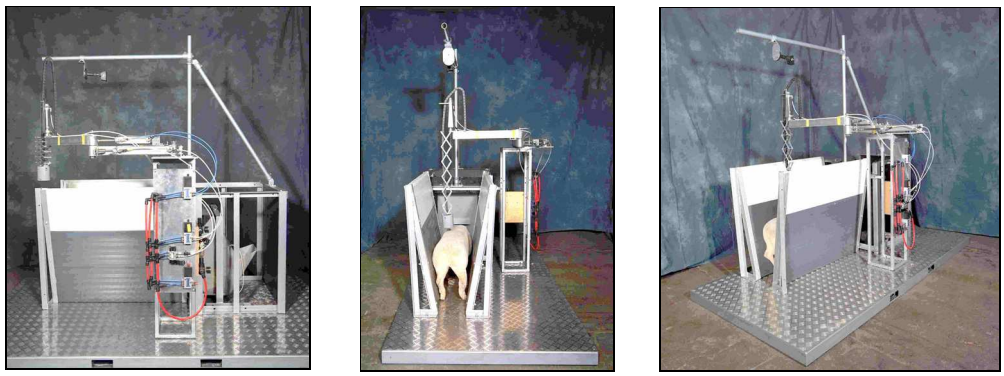


Figure 3.16 Three views of the robot feeding rig out of situe. Silsoe Research Institute.

The robot was actually constructed in work previous to this thesis; however, it had never before been experimentally tested in a real environment or controlled by image analysis software. The robot was of a two-axis SCARA (Selective Compliant Assembly Robot Arm) design. The obvious alternative would have been a Cartesian arrangement. However, it was thought the necessary rails would have been liable to dust accumulation, and would also have been untidy as the SCARA arrangement can retract to the side when not in use (Frost et al. 2004).

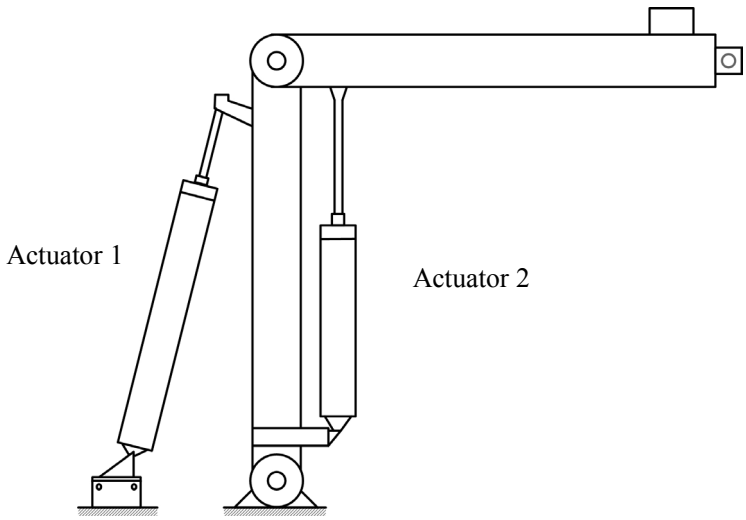


Figure 3.17 Plan view of the arrangement of the actuators. Actuator 1 has length of 352mm when closed and a stroke of 200mm, actuator 2 has a closed length of 295mm and a stroke of 100mm.

The control box for the robot was also constructed prior to this work, and was operated by sending coordinates in the robot frame using an RS-232 interface on the control PC.

3.6.2. Calibration and integration with image analysis system

The coordinates calculated by the image analysis software are used to control the robot location. These coordinates need to be converted to the frame of reference of the robot so that the control software for the actuators can interpret them correctly.

The robot had previously been calibrated to move to locations on its horizontal plane of movement as specified by its internal control software. This pre-calibration was integrated with the analysis software using a variation of the Table and Grid method (Trevelyan 2004), to map image coordinates to robot coordinates. A horizontal grid was measured out on a horizontal plane and placed under the robot laser pointer on the end of the robot arm. The robot was instructed to move to certain coordinates, and the corresponding error was measured between the resting position of the laser spot and the target position on the grid. Effectively this problem was not one of calibrating the robot itself, but instead estimating the offsets and scaling factors to convert between robot co-ordinates and image coordinates. Full robot calibration is a complex task and one not explored by this work.

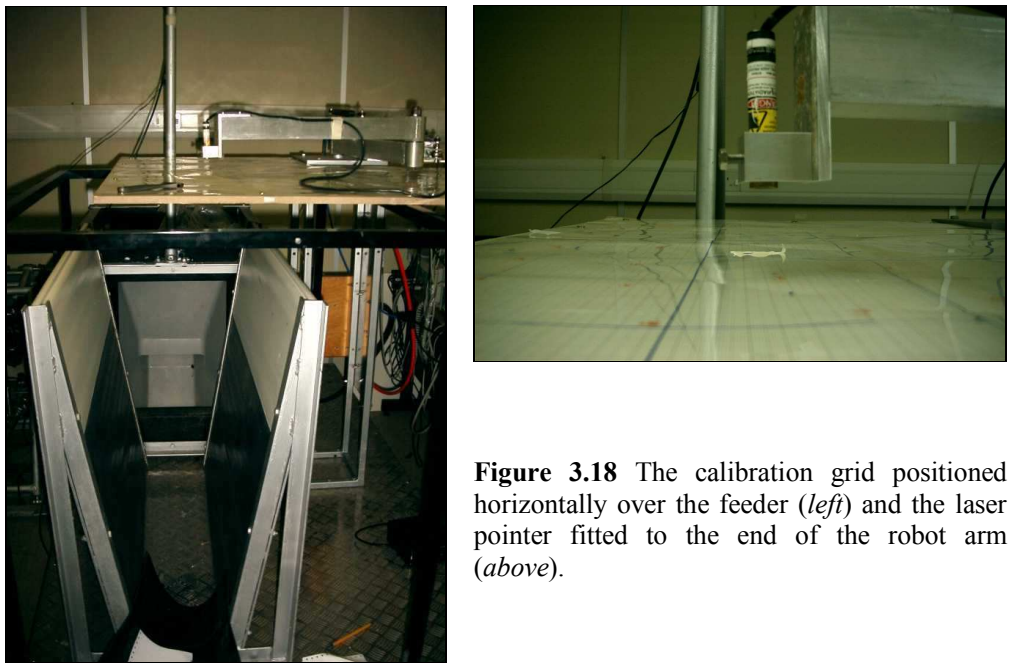


Figure 3.18 The calibration grid positioned horizontally over the feeder (*left*) and the laser pointer fitted to the end of the robot arm (*above*).

The overhead camera has a coordinate system that is independent from that of the robot control system. However, the plane of the image in the camera view is very nearly the same as the plane of operation of the robot, i.e. they can both be considered horizontal. Therefore, the conversion from the image plane to the robot coordinate system can be approximated as a simple translation of the image coordinates, and a scaling of the axes. The robot was calibrated before being moved into place in the animal house. After calibration, the lens settings and location of the camera were locked into place. Once the robot rig was in its final position, a spot-board was used to check that the robot was behaving as expected – see Figure 3.19. This was achieved by directing the robot to spots marked on a board placed in the feeder at a typical pig height. The robot was directed to the spots by clicking on them in the image analysis system.

Figure 3.19 Example of a spot board testing of the positional accuracy of the robot in its final position. The spot board is placed at expected height of a pig and the user clicks on the white tape markers (20mm x 20mm) within the image analysis software. The robot should move such that the laser falls within the tape outline. The laser position has been highlighted for clarity. Mounted on the end of the robot arm are the observation camera and dust cover, and the laser.

It is important to know the number of real world units per pixel as height varies, so that error values in pixels may be converted to real world units. This was calculated using a calibration board with known dimensions marked on it. This board was placed at different heights in the pen, and then the number of pixels between the markings in the captured images from the boom camera was calculated using Photoshop.

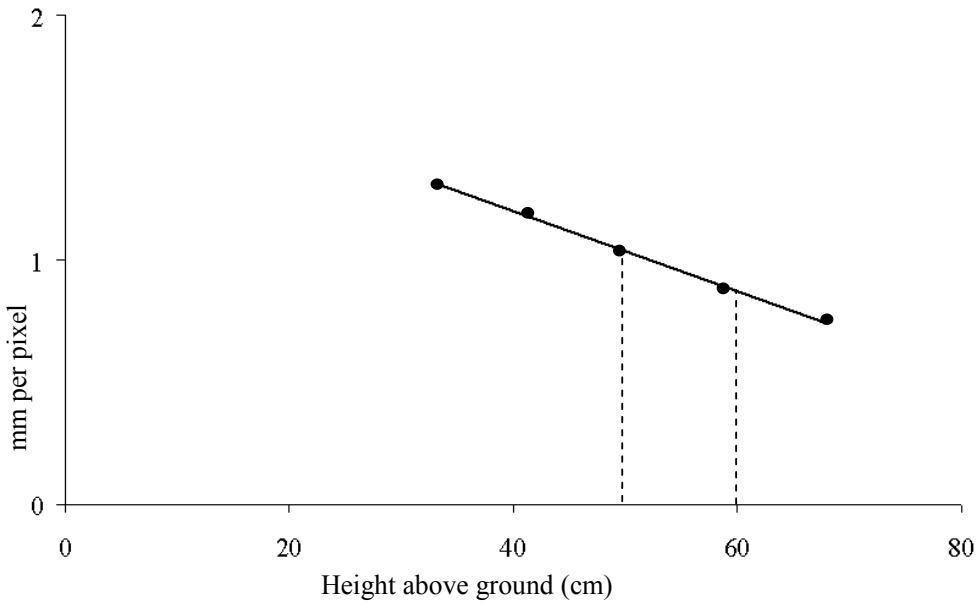


Figure 3.20 Graph showing linear regression line (R^2=0.99) of millimetres per pixel against height above ground in images taken from the boom camera. Vertical lines indicate pig height limits of the animals in this work.

3.7. Experiment: Determining system success

3.7.1. Environment

12 Duroc X Landrace X Large White females were available for the trials. They were raised from 3 weeks in an animal house at Silsoe Research Institute. Experiments were carried out in the environment in which they lived. The animals were approximately 13 weeks old at the time of the trials, and weighed 56.3 kg on average. Ambient daylight entered through a number of small, high windows, and was accompanied by a number of artificial lights on the ceiling.

3.7.2. Equipment

The robotic feeding station was placed in the animal house so that the animals had continual free access to it. The pigs were raised from 3 weeks of age with the feeder in the pen, and regular food being provided in both the robotic feeder and the conventional feeding troughs. The feeding station was illuminated by a combination of ambient daylight and electric lights, and from a 60w bulb above the feeder itself. The robot was only functional when an operator was present.

Figure 3.21 Photo of the working environment, showing the robot rig and some pigs.

3.7.3. Procedure

The aim of a trial was to see how well the image analysis and robot placement systems performed while a pig was feeding in the rig. In order to achieve results that were as natural as possible, the animals were not forced into the feeder. Trials were run when animals came to feed of their own free will. The activations were recorded over two approximately one-hour windows for each of two days. Visits where a pig walked into the feeder and stayed only for a short period of time were discounted, as were incomplete visits (where a pig was already in the feeder, for example). This left 7 complete 'visit sessions' to be analysed, each consisting of the time from when a pig enters the feeder to when it leaves, and all the activations of the robot as it fed. As the pigs entered the feeder during the trials, the target P2 position was marked on their backs by manually placing a square of black tape measuring approximately 25mmx25mm at the correct location. This location was determined using information in prior literature (see section 3.3.1 for more information), and the tape was placed in the presence of a skilled pig handler. This mark was to enable the error between the predicted

position of the software and the actual P2 position as indicated by the marker tape to be later measured. On some occasions, a pig may have entered the stall but only remained there for a few seconds. These occasions were discounted from the trials, as the robot has no chance to activate. If the robot were entirely autonomous, this situation would happen often and would not be a problem.

Once a pig enters the stall to feed, the image analysis software begins tracking the animal (see 3.5.2). The user can then manually set the *a* and *d* offset constants in the model - see equations (2) and (3). These offsets vary from animal to animal, as described previously. In the future, if the animals were tagged electronically these parameters could be calibrated once, and then recalled by retrieving information for particular pigs as a tagged animal enters the feeder. For now, this is done manually, and is made as intuitive a process as possible in the software by allowing the user to click on the video display window to set the offsets. The system then automatically detects when the animal has been stationary for a set length of time, and then activates the robot. The robot is normally then reset when a pig is deemed to have moved, based on the motion of the rump. However, for this work the threshold to detect animal movement was increased to make the system less sensitive to movement post-robot activation. This was to allow the error in position to be recorded over a longer period of time.

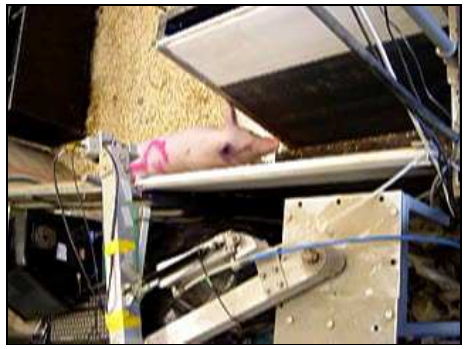
The observation camera mounted on the end of the robot arm next to the laser pointer is connected to a VCR and a recording is made of all activations. These recordings are later used to measure the positional error between the laser spot and the real P2 position as marked by the tape.

3.8. Results

The results of the sensor placement experiment are now presented. Some related results have been published by the author at conferences and in journals (French et al. 2003; Frost et al. 2004).

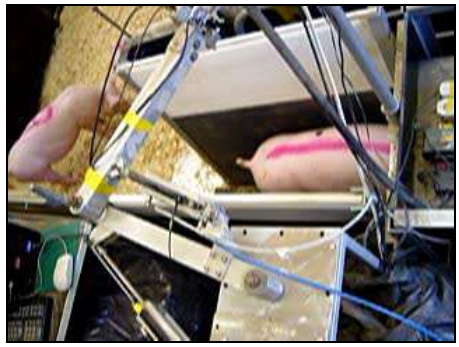
The flow of events that take place in a visit to a feeder is presented in Figure 3.22.

1. Pigs outside feeder



2. Pig entering feeder

3. Pig eating



4. Robot activates, normally several times

5. Pig exits and robot resets

Figure 3.22 Storyboard of an animal's visit to the feeder.

Graphs are presented to summarise results of the accuracy measures. It was hypothesised that the sensor deployment mechanism would take about one second to deploy after robot arrival. This is consistent with previous work (Frost et al.

2000) which hypothesised a window of two seconds to take a reading, including robot deployment time. The robot takes the order of a second to finish moving (depending on the air pressure delivered to the pneumatics and amount of stiction in the joints), and so allowing an extra second after this to deploy the sensor is reasonable. Therefore, the error graphs represent the error that would be present in the placement if a reading was taken one second after the robot had finished moving (see Figure 3.23).

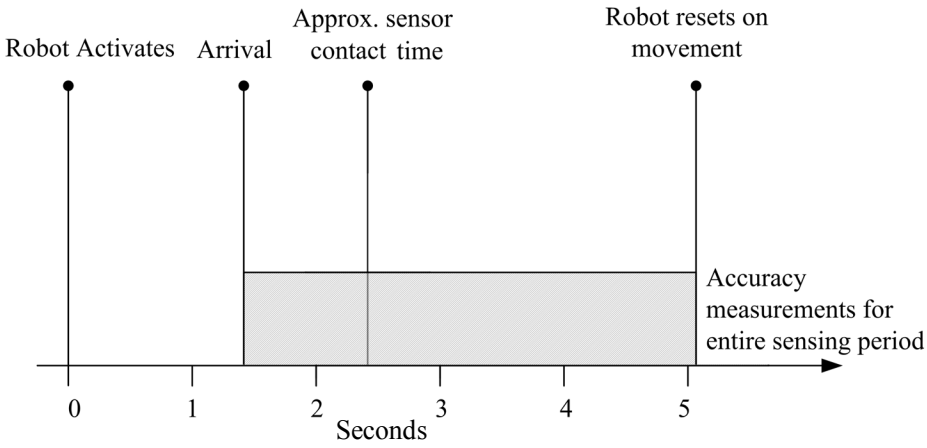


Figure 3.23 Diagram to illustrate the timescales of the measurements. The "Sensor contact time" marker is where the placement errors are measured.

Seven visits to the feeder were analysed. A visit consists of a pig entering the feeder, and the robot activating one or more times while the pig feeds, before finally the pig exits the rig. These seven visit sessions ranged in time from one minute to eleven minutes. The mean length of time spent in the feeder was 4 minutes and 51 seconds. Out of these seven visits, five different pigs entered the feeder; two of them were repeated visits by the same pig (one pig visited on both visits 1 and 3, and another pig on both 5 and 7). With a total of 34 minutes of visits and 75 activations, it can be estimated that a robot activation can take place approximately every 30 seconds that a pig is in the feeder.

Errors from the image analysis system and model are presented now, from frames saved when activation commands are sent to the robot. Therefore, they represent the state of the tracking system and model P2 estimation when drive commands

are sent to the robotic control system, *before* any robot inaccuracies are introduced.

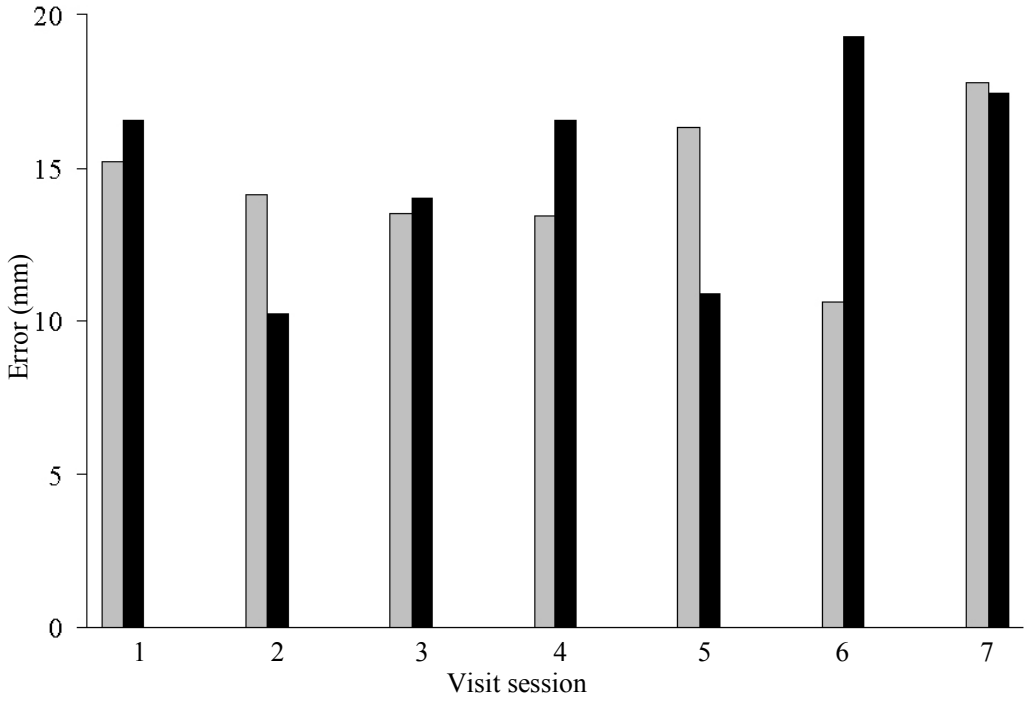


Figure 3.24 Graph indicating the kink location RMS errors (grey) and the P2 prediction RMS errors (black) across the activations in each visit sequence.

The kink location errors (grey bars, Figure 3.24) were calculated by measuring the error on each individual kink location manually, from the frame saved at the point of robot activation. Examples of these images are the 'Activation' images that will be presented in the main results, Section 3.8.1. The P2 prediction error (black bars, Figure 3.24) is RMS of the Euclidean distances between the model-predicted P2 point and the physical P2 marker on the animal, again at the point of robot activation.

Notice that the P2 model errors in Figure 3.24 are sometimes less than the kink location errors for that sequence. This may be for a number of reasons. First, the errors may be on lower-weighted points of the model and so have less effect. Second, some of the errors may cancel out, for example the back kinks erroneously moving forward and the forward kink moving back together would locate the P2 point in a similar position than if no movement had occurred.

The P2 prediction error is the most important as far as the final system is concerned, as this is effectively the error on the input given to the robotic system. The mean P2 prediction error is ~15mm across the seven sessions. This is more than the 6-8mm predicted (Tillett et al. 2002) when using a linear model with separate offsets. This may be due to a number of factors. The original model (Tillett et al. 2002) used six feature points versus the model used here which has four, and although the four-point model was shown to be of comparable accuracy with test data, perhaps in the actual experiments this new model did have a negative effect on accuracy. Also the set of pigs used to train the model may in some way have a different geometry to the animals used in this work. Some errors are bought in from the kink location errors (grey bars, Figure 3.24), and are then propagated through the P2 model to form a part of the error present in the final prediction. However, the overall prediction accuracy of 15mm falls within the specified limit of 20mm error, leaving a 5mm buffer for robotic system errors.

Knowing *which* kink points introduce the most error would allow the system to associate reliability scores with each kink coordinate. This in turn might allow the model to be more robust to ambiguous kink location on frames where a clear kink feature is not visible. It was considered that this would not be necessary for this initial system, but it is recognized that this may be one future method of reducing the error levels in the P2 prediction from the kink locations, if some of the kink points were found to be more reliable features than others.

Section 3.8.1 will go on to consider the final error in the system, where the robotic placement is compared to a human-set marker point at the P2 position.

3.8.1. Graphs of robot position error (mm) from the target P2 position.

Over the following pages, results will be presented from the real-life application of the sensor location system on feeding pigs. For each of the seven different visits to the feeder that were analysed, three types of results will be presented. First, graphs of sensor location error will be presented. These are the longitudinal, lateral and Euclidean distance errors present between the robot's final position, as indicated by the laser spot, and the P2 point as manually marked on the animal. They are measured approximately one second after the robot has completed its movement as directed by the image analysis software. These errors represent the accuracy of the complete sensor location system, and from these the average placement error and the number of sufficiently accurate readings per day can be estimated.

The second kind of results to be presented are example frames from the two overhead cameras. An example frame from the image analysis software at the point of robot activation is presented first, followed by a frame from the boom camera taken at the corresponding point of robot arrival. The first of these images allows the reader to see the state of the image analysis components and the model estimate of P2 position at the point of activation. The boom image shows the locations of the actual P2 point compared to the robot position at the time the robot arrives at its final position. The particular images were chosen because they are good examples of either representative or unusual situations.

These images are then followed by some graphs which show some typical and atypical example activations during the animal's visit to the feeder. They are chosen because they are interesting or typical cases, and are accompanied by individual explanations. On these graphs, the robot is activated at zero seconds, and arrives at its final destination (i.e. the modelled P2 point) when the error plotting begins. The movement of the rump is plotted from the point the robot activates.

Finally, Figure 3.46 presents summary results for the Euclidean distance between the laser spot and P2 marker for every activation analysed. This is overlaid with an estimated 'adjusted error' which is theoretically possible to achieve by taking tracked rump information into account, basically by accounting for longitudinal rump movement in the longitudinal error of the placement. By incorporating the rump motion that takes place up until the sensor is actually in place over the estimated P2 position, the longitudinal error present in the placement can often be reduced.

A discussion of these results is presented in section 3.9.

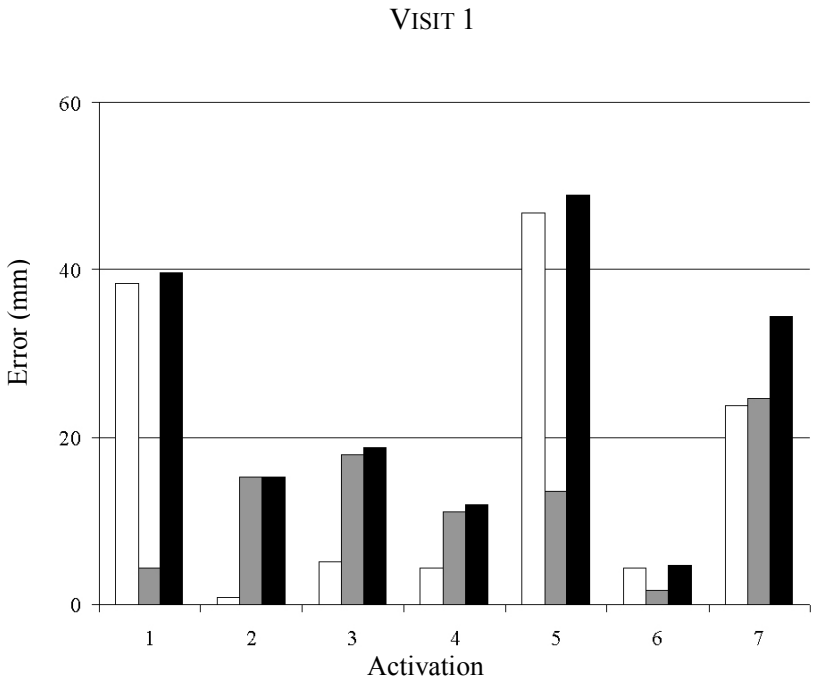
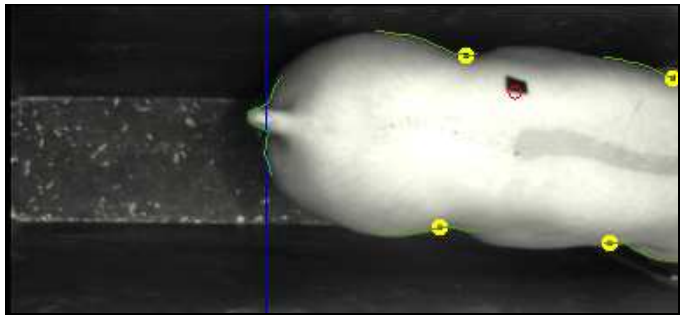
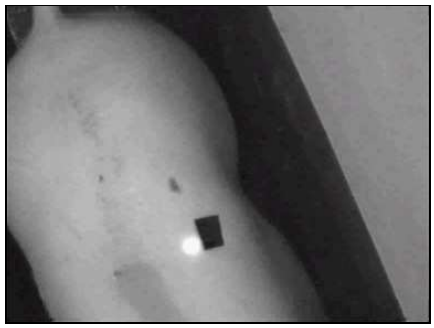


Figure 3.25 Placement error graph, visit 1. Errors between the robot sensor position as marked by the laser spot and the P2 mark on the pig's back. White = longitudinal error, grey = lateral error, black = Euclidean distance error.



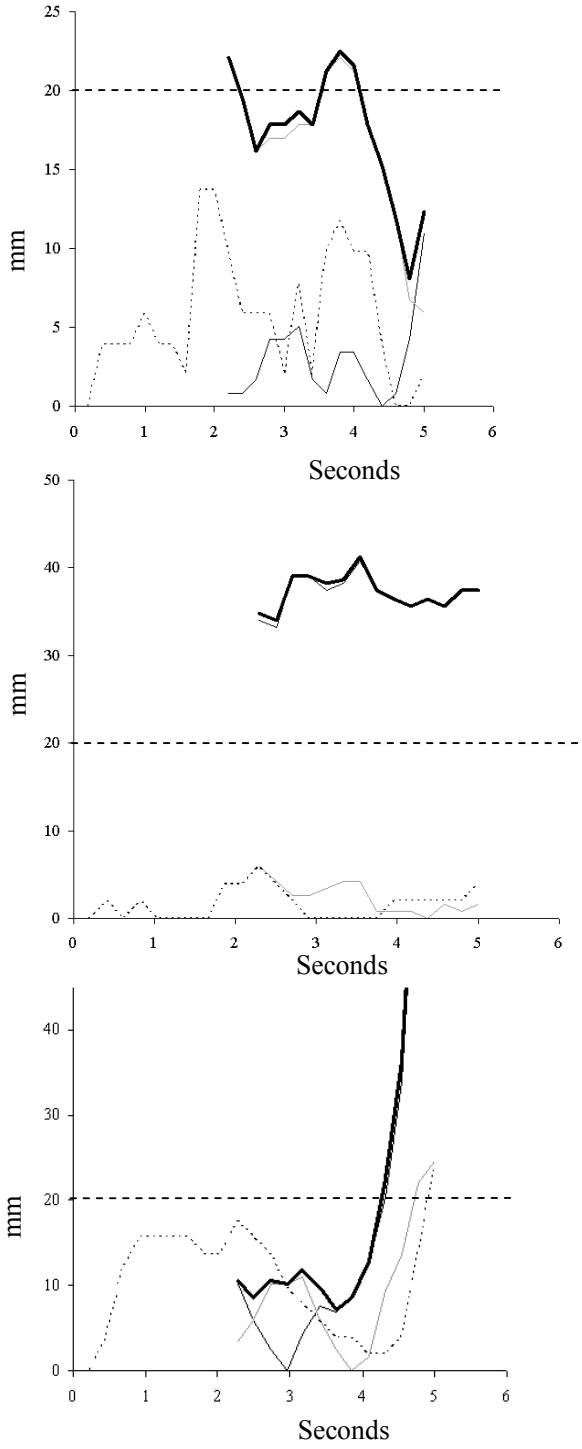
Activation image, activation 3. Note how the image processing has correctly located the kink points, despite the angle of the pig. Note also that the P2 model has accurately located the P2 point.



Boom image at robot arrival, activation 3. The pig may have moved slightly since activation. The laser point is about 19mm from the centre of the P2 marker.

Figure 3.26 Example images. (Top) Typical activation frame output by the image analysis software at the point of robot activation, and **(bottom)** a boom camera frame of the resulting position of the laser when the robot is in position for activation 3.

GRAPHS SHOWING P2 POSITION ERRORS AND RUMP MOVEMENTS FOR SOME ACTIVATIONS



Activation 3

Note how most of the error is lateral. This is likely to be because the pig is leaning to one side, and therefore does not fit the built model well. This can be seen in Figure 3.26.

Activation 1

Most of the error is longitudinal, but the pigs rump does not move very far at all. This error is likely to be caused by the pig hunching up, or stretching out, after the robot has been activated.

Activation 4

A good example of the system in a typical situation. The errors are low for a few seconds, and then the pig backs out of the feeder. The robot would have enough time to activate and take a reading before the error becomes too large (error on sensor contact is 11.9 mm)

Figure 3.27 Graphs depicting a typical activation event and some atypical scenarios during visit 1. The bold line is the Euclidean distance error between the laser and the marked P2 position. The thin grey line is the lateral laser-P2 error and the thin black line is the longitudinal laser-P2 error. The dotted line represents the longitudinal movement of the pig's rump throughout the sequence. The error lines are not plotted until the robot has reached its final destination. The dashed horizontal line represents acceptable error.

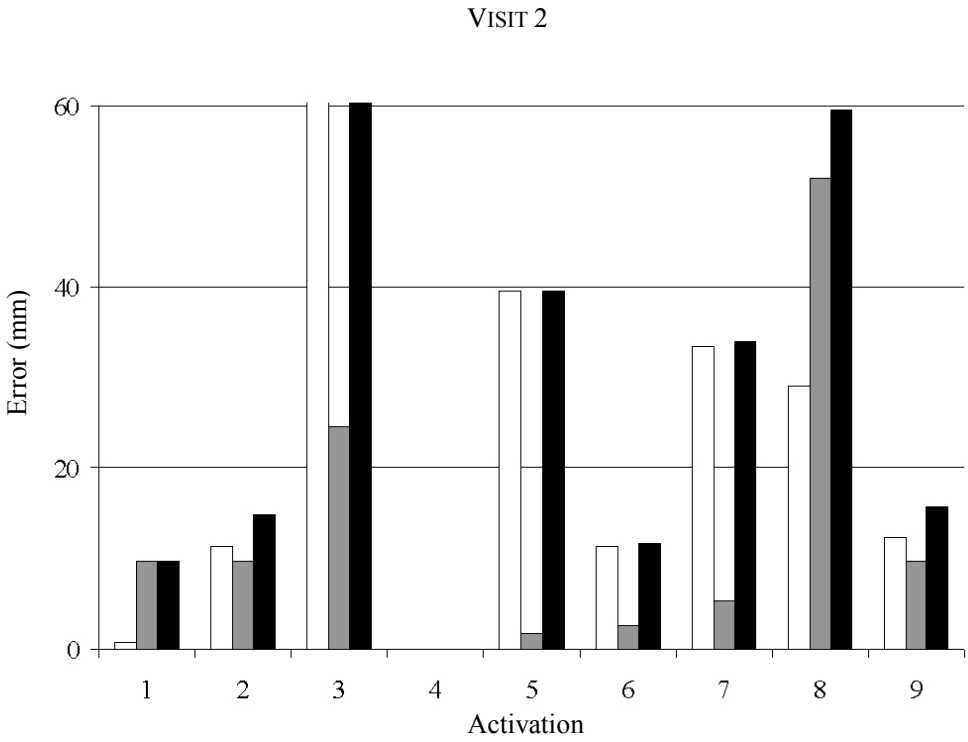
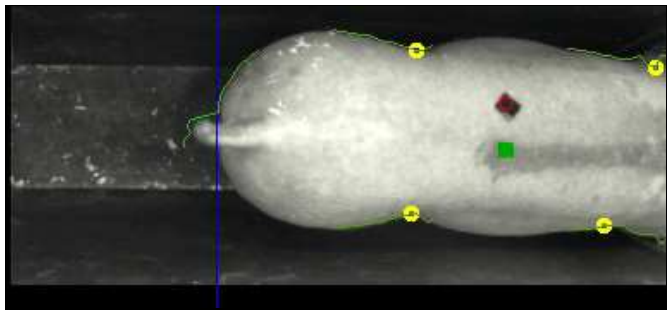


Figure 3.28 Placement error graph, visit 2. Errors between the robot sensor position as marked by the laser spot and the P2 mark on the pig's back. White = longitudinal error, grey = lateral error, black = Euclidean distance error. The missing data is generated by immeasurably large errors – this is explained in the discussion.

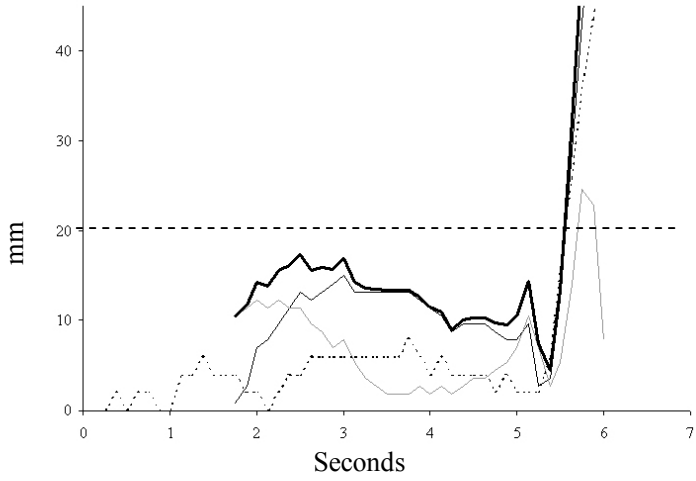


Activation image, activation 9. Good location of kink points and good P2 location from model. Note the potential for the tail to change its perceived location.

Boom image at robot arrival, activation 9 Error of approximately 10mm between the laser and the centre of the marker tape on robot arrival.

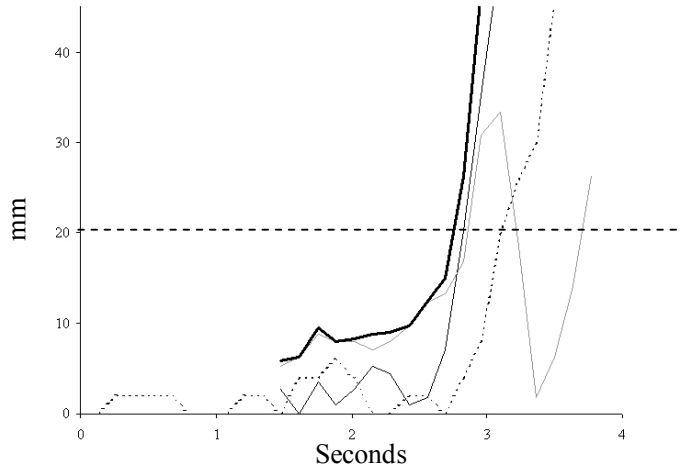
Figure 3.29 **Example images.** (Top) Typical activation frame output by the image analysis software at the point of robot activation, and (bottom) a boom camera frame of the resulting position of the laser when the robot is in position for activation 9.

GRAPHS SHOWING P2 POSITION ERRORS AND RUMP MOVEMENTS FOR SOME ACTIVATIONS



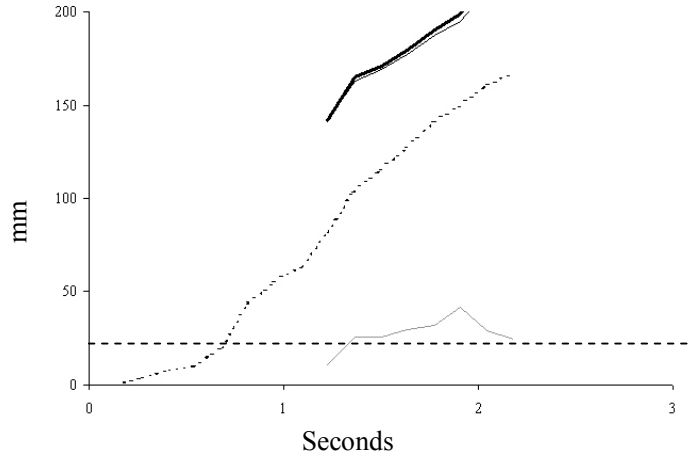
Activation 9

The error is within the 20mm limit for the sequence, except at the end where the pig backs out, causing the robot to reset.



Activation 1

Here the pig moves out shortly after activation. There is still enough time for a sensor contact though (hence low contact errors in Figure 3.28)
Note the lateral 'wobble' present on this graph as the pig backs out. This is caused by the characteristic rocking motion of pigs.



Activation 3.

A situation caused by the pig exiting back directly after robot activation. This could be caused by pneumatic noises of the robot.

Figure 3.30 Graphs depicting a typical activation event and some atypical scenarios during visit 2. The bold line is the Euclidean distance error between the laser and the marked P2 position. The thin grey line is the lateral laser-P2 error and the thin black line is the longitudinal laser-P2 error. The dotted line represents the longitudinal movement of the pig's rump throughout the sequence. The error lines are not plotted until the robot has reached its final destination. The dashed horizontal line represents acceptable error.

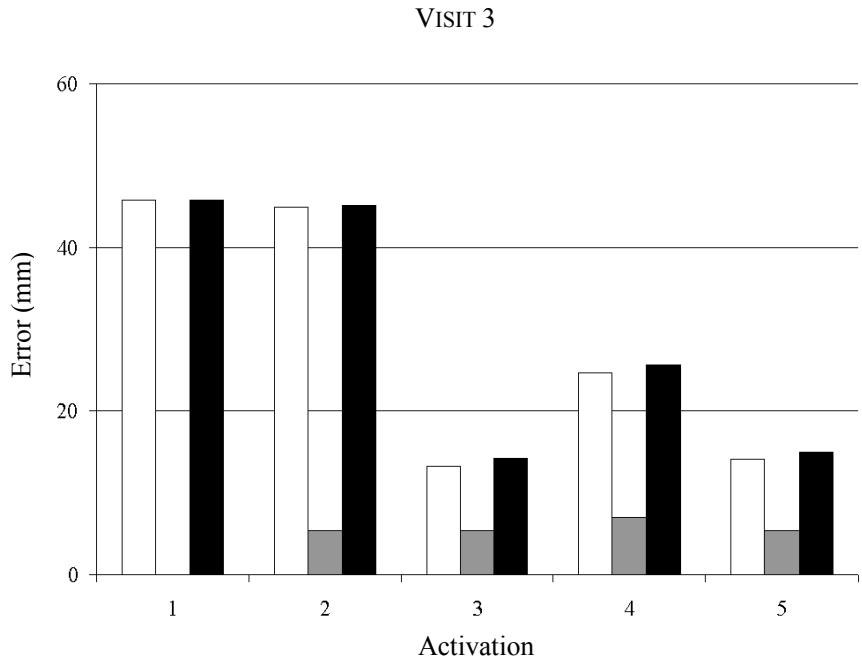
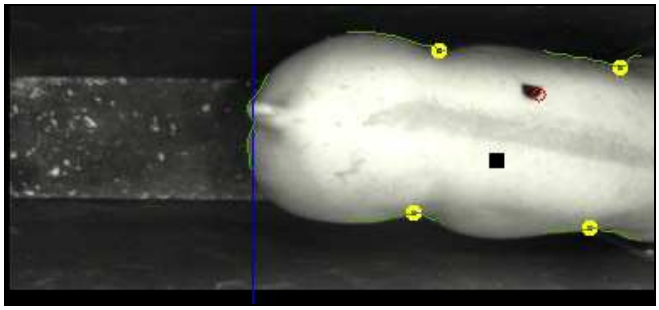


Figure 3.31 Placement error graph, visit 3. Errors between the robot sensor position as marked by the laser spot and the P2 mark on the pig's back. White = longitudinal error, grey = lateral error, black = Euclidean distance error.



Activation image, activation 5.

Note how the P2 prediction is not perfect, despite accurate locations of kink points.

Boom image at robot arrival, activation 5.

The robots final position is further back than the goal position seen in the figure above.
This is because the pig moves along the feeder between activation and arrival (as can be seen in Figure 3.33 below).

Figure 3.32 Example images. (Top) Typical activation frame output by the image analysis software at the point of robot activation, and (bottom) a boom camera frame of the resulting position of the laser when the robot is in position for activation 5.

GRAPH SHOWING P2 POSITION ERRORS AND RUMP MOVEMENTS FOR SOME ACTIVATIONS

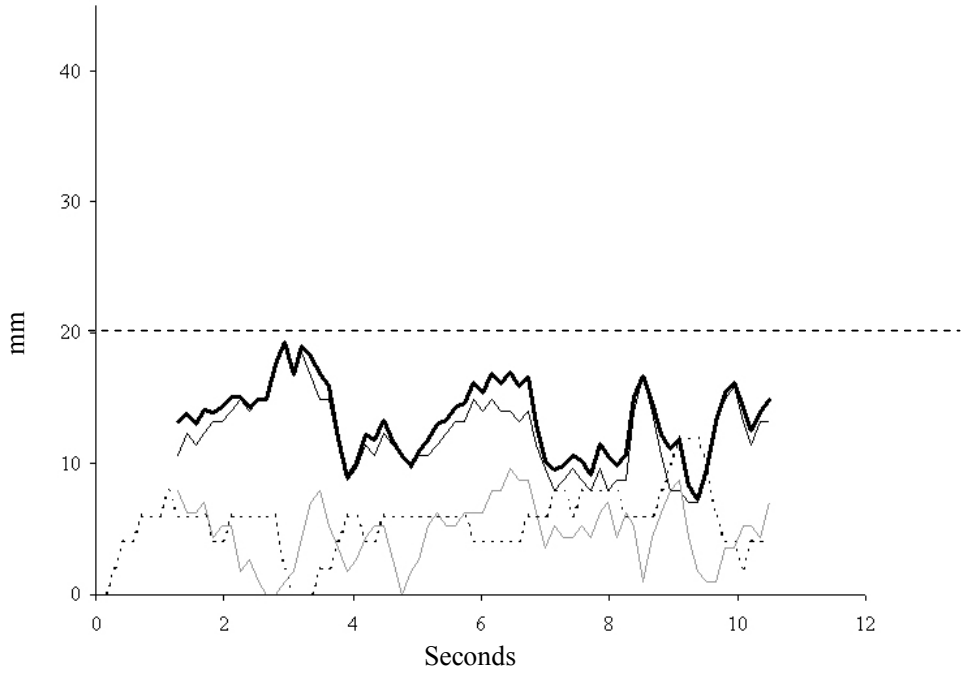


Figure 3.33 Activation 5 The bold line is the Euclidean distance error between the laser and the marked P2 position. The thin grey line is the lateral laser-P2 error and the thin black line is the longitudinal laser-P2 error. The dotted line represents the longitudinal movement of the pig's rump throughout the sequence. The error lines are not plotted until the robot has reached its final destination. The dashed horizontal line represents acceptable error**.**

This graph shows most of the error present being caused by longitudinal motion of the pig. Because of the error sometimes present in the model prediction of the P2 point, longitudinal offset of the pig from its activation position can actually *lower* the overall error: note how when the rump location moves at about 2.7 seconds on Figure 3.33, the longitudinal and combined error actually *falls*. This is because although the animal has moved, it has moved *nearer* to the target P2 position. However, note also that the errors for this sequence are low (within accuracy); this tends to be a small effect.

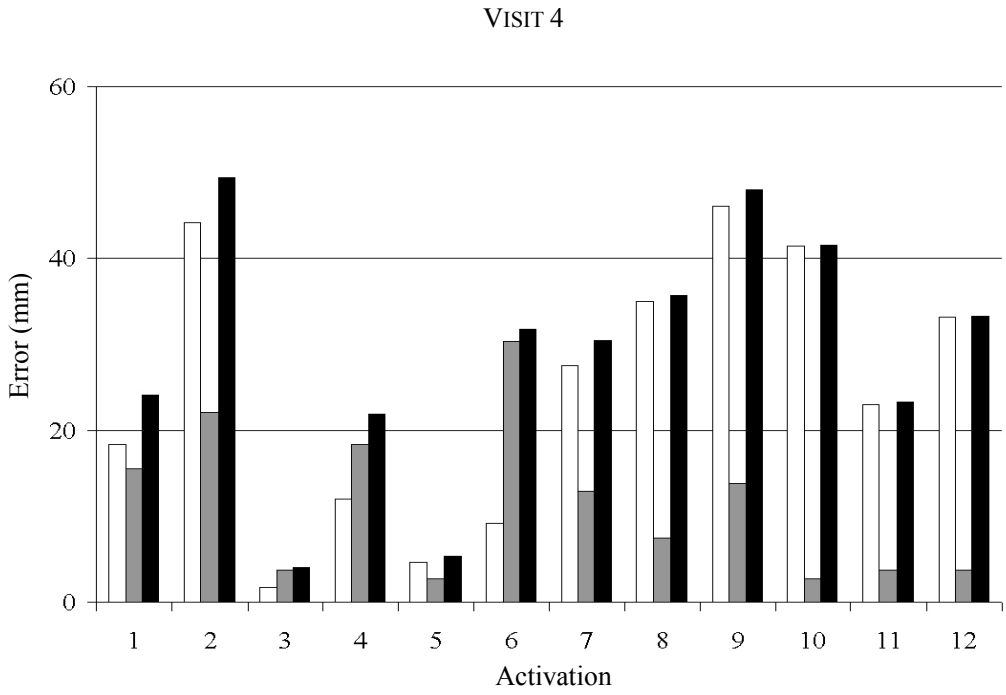
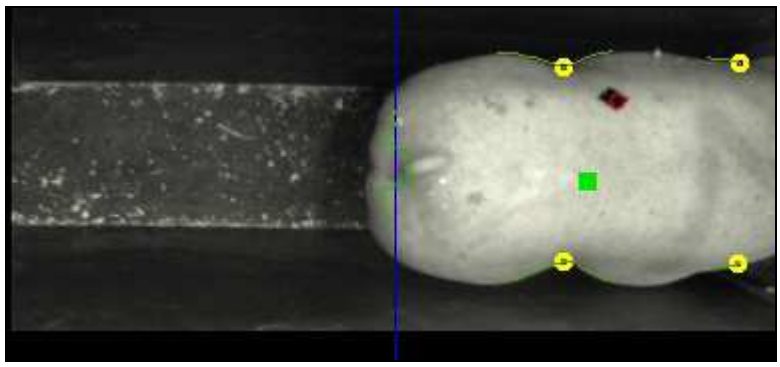
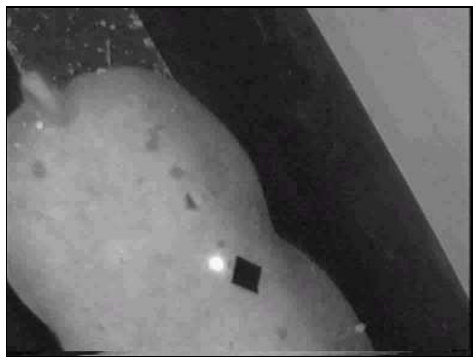


Figure 3.34 Placement error graph, visit 4. Errors between the robot sensor position as marked by the laser spot and the P2 mark on the pig's back. White = longitudinal error, grey = lateral error, black = Euclidean distance error.



Activation image, activation 1. Good P2 prediction despite location of ambiguous front-right kink point

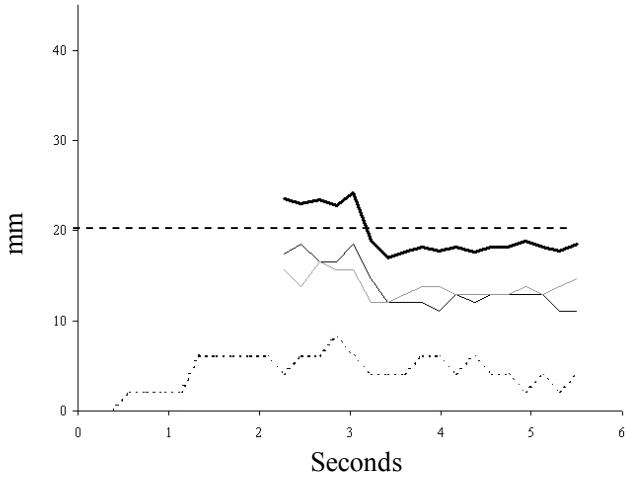


Boom image at robot arrival, activation 1.

The laser point is offset because the pig moved in the feeder as the robot activated

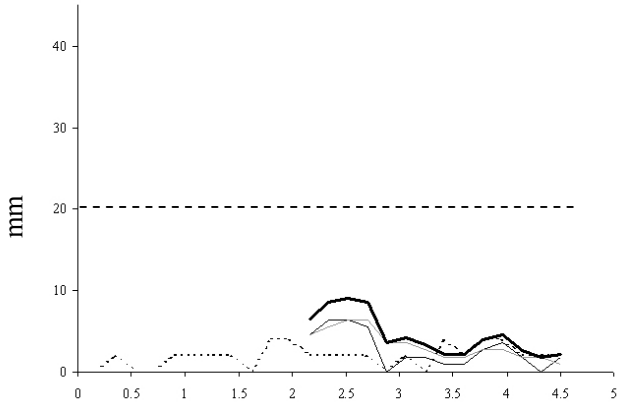
Figure 3.35 Example images. (Top) Typical activation frame output by the image analysis software at the point of robot activation, and (bottom) a boom camera frame of the resulting position of the laser when the robot is in position for activation 1.

GRAPHS SHOWING P2 POSITION ERRORS AND RUMP MOVEMENTS FOR SOME ACTIVATIONS



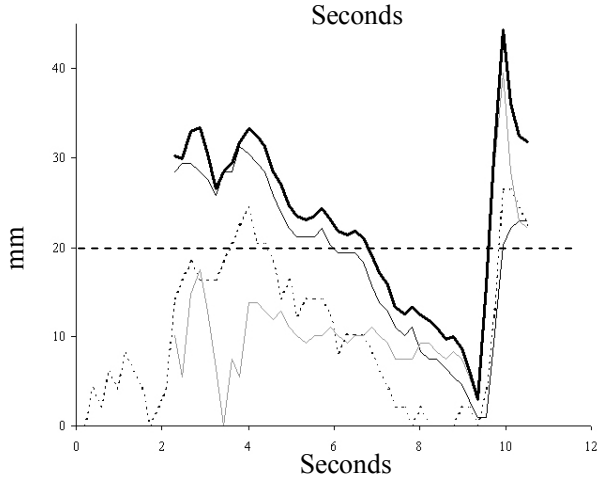
Activation 1

The error rises and falls as the pig shuffles forwards and backwards in the feeder. Note that the error falls as time goes on. A system that made use of repeated sensor applications could make use of this, assuming fat thickness falls as positional error falls.
One reason the error is high initially is because the pig has moved during robot activation, as can be seen from the rump motion line (see Figure 3.35 for the consequence of this)



Activation 3

Example of small errors where a pig stands still and eats.



Activation 7

Longitudinal motion is accompanied by lateral 'wiggle'. Note how rump movement is a good indication of overall error.

Figure 3.36 Graphs depicting a typical activation event and some atypical scenarios during visit 4. The bold line is the Euclidean distance error between the laser and the marked P2 position. The thin grey line is the lateral laser-P2 error and the thin black line is the longitudinal laser-P2 error. The dotted line represents the longitudinal movement of the pig's rump throughout the sequence. The error lines are not plotted until the robot has reached its final destination. The dashed horizontal line represents acceptable error.

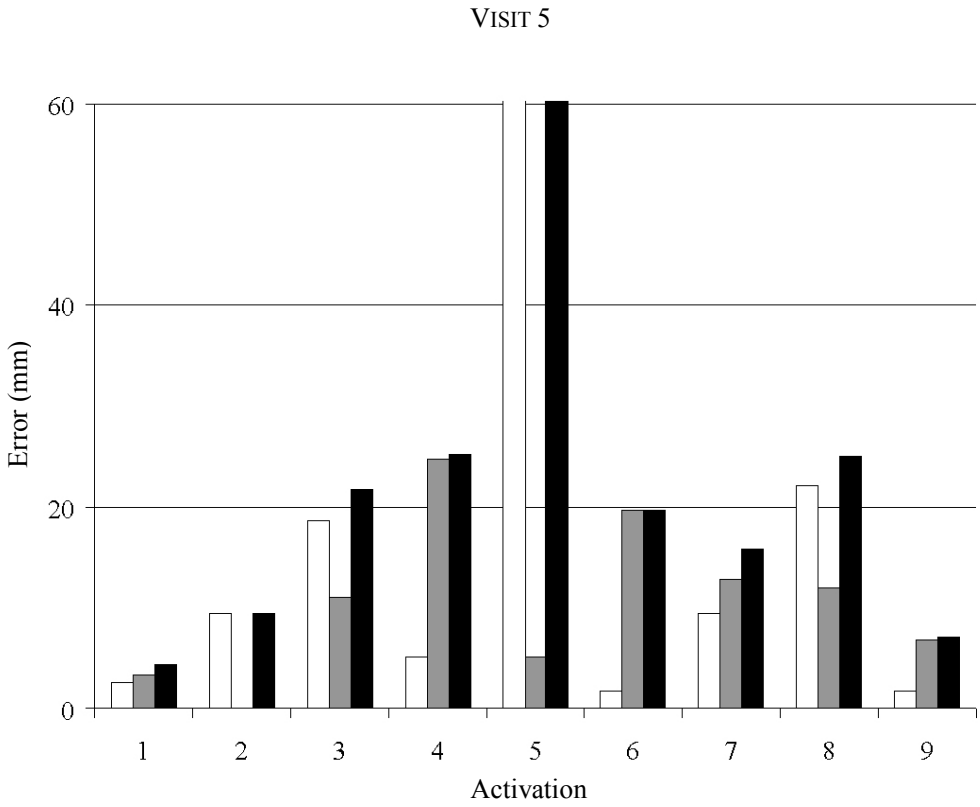
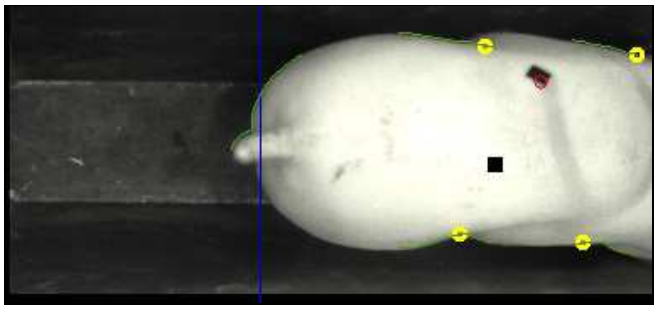
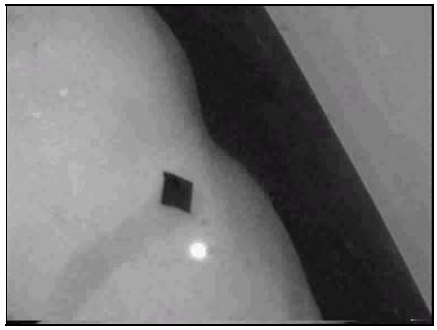


Figure 3.37 Placement error graph, visit 5. Errors between the robot sensor position as marked by the laser spot and the P2 mark on the pig's back. White = longitudinal error, grey = lateral error, black = Euclidean distance error.



Activation image, activation 5.
Good kink locations and P2 modelling, despite vague front-left kink.



Boom image at robot arrival, activation 5.

The animal has already moved back. This can be seen in Figure 3.39.

Figure 3.38 Example images. (Top) Typical activation frame output by the image analysis software at the point of robot activation, and (bottom) a boom camera frame of the resulting position of the laser when the robot is in position for activation 5.

GRAPHS SHOWING P2 POSITION ERRORS AND RUMP MOVEMENTS FOR SOME ACTIVATIONS

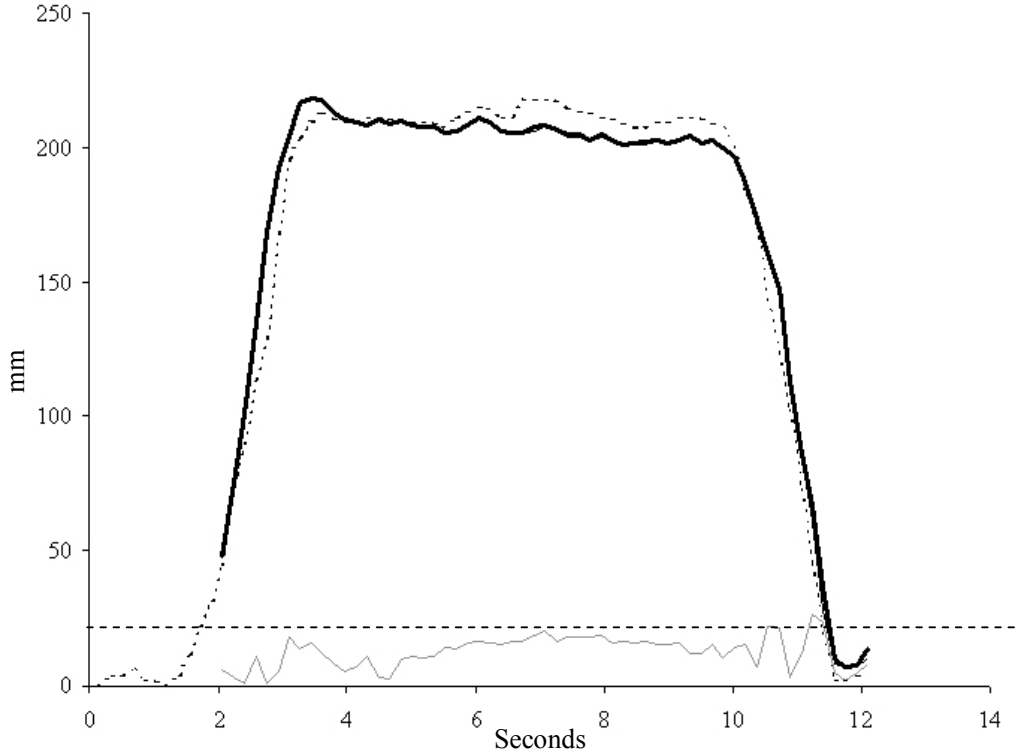


Figure 3.39 Activation 5. The bold line is the Euclidean distance error between the laser and the marked P2 position. The thin grey line is the lateral laser-P2 error and the thin black line is the longitudinal laser-P2 error. The dotted line represents the longitudinal movement of the pig's rump throughout the sequence. The errors lines are not plotted until the robot has reached its final destination. The dashed horizontal line represents acceptable error**.**

Figure 3.39 illustrate the large errors present when a pig takes a step back from the feeding trough, followed about 10 seconds later by a step forward again.

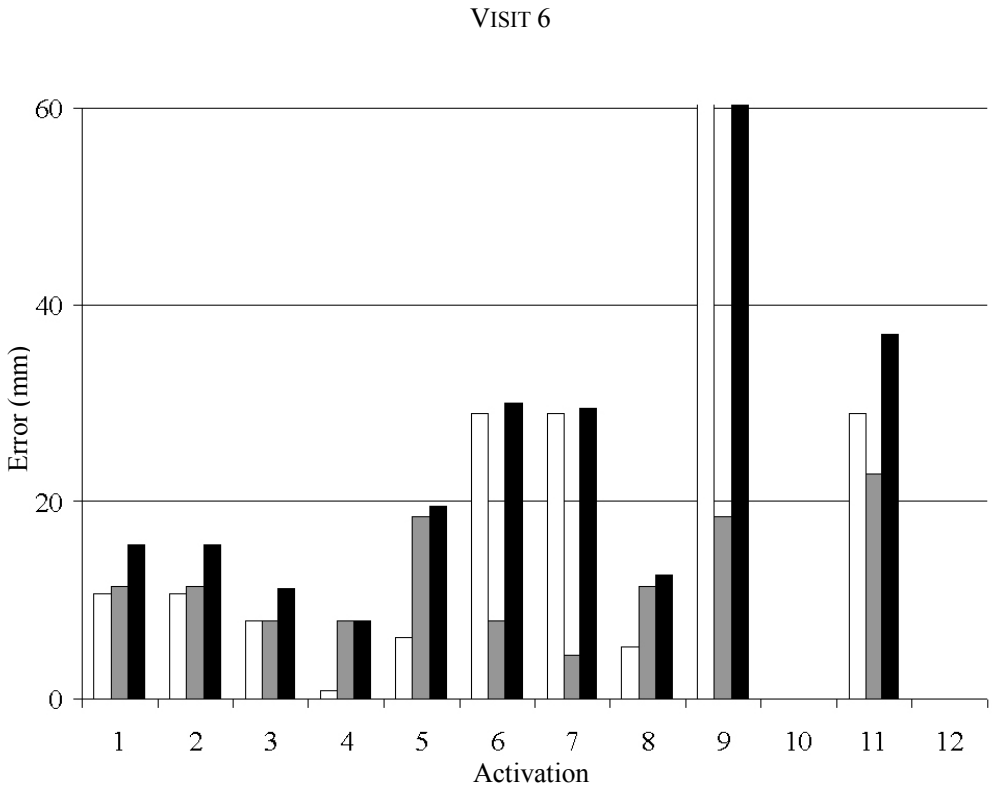
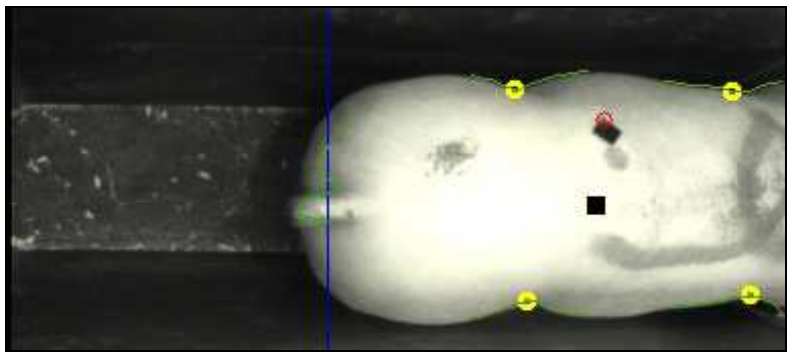


Figure 3.40 Placement error graph, visit 6. Errors between the robot sensor position as marked by the laser spot and the P2 mark on the pig's back. White = longitudinal error, grey = lateral error, black = Euclidean distance error.

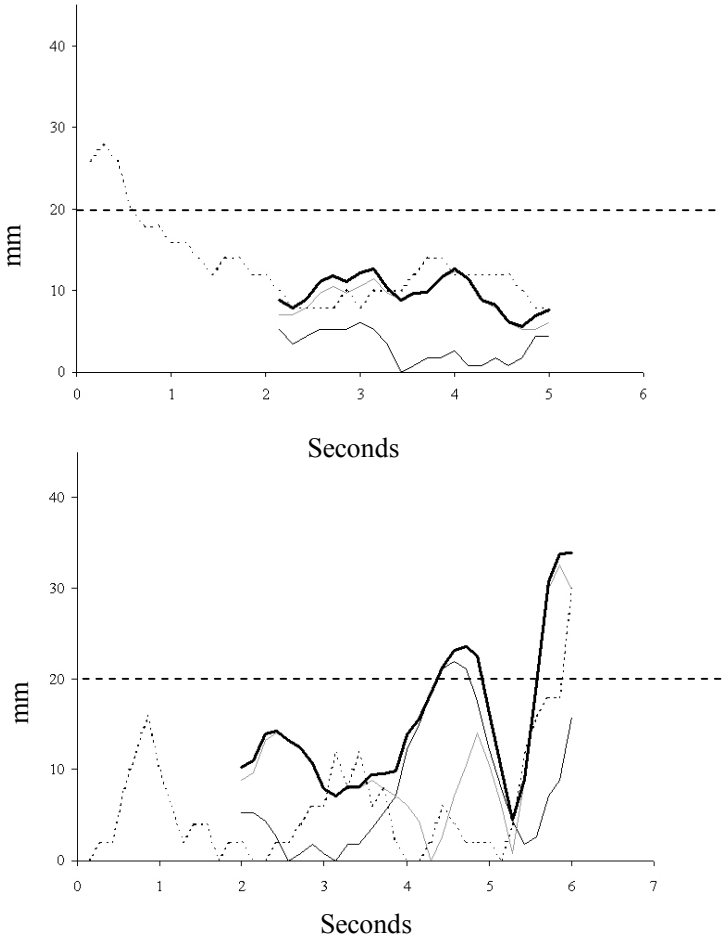


Activation image, activation 8. Non-perfect rump location still allows the kink detectors to be placed satisfactorily.

Boom image at robot arrival, activation 8.

Figure 3.41 Example images. (Top) Typical activation frame output by the image analysis software at the point of robot activation, and (bottom) a boom camera frame of the resulting position of the laser when the robot is in position for activation 8.

GRAPHS SHOWING P2 POSITION ERRORS AND RUMP MOVEMENTS FOR SOME ACTIVATIONS

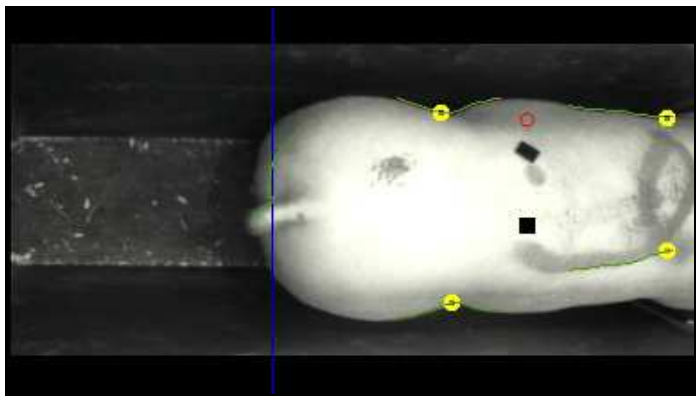


Activation 8

Note the relatively high proportion of lateral movement (hence the lateral offset in positional error, visible in the boom camera in Figure 3.41.)

Activation 4

Reasonable error values despite the longitudinal motion, and the lateral 'wobble' that accompanies this motion.



Activation 11

Example of tracking error of the front right boundary detector. Note how the detector is stuck on clutter, and the resulting error in the P2 model prediction.

Figure 3.42 (*top*) Graphs depicting some example scenarios during visit 6. The bold line is the Euclidean distance error between the laser and the marked P2 position. The thin grey line is the lateral laser-P2 error and the thin black line is the longitudinal laser-P2 error. The dotted line represents the longitudinal movement of the pig's rump throughout the sequence. The error lines are not plotted until the robot has reached its final destination. The dashed horizontal line represents acceptable error.
(*bottom*) The bottom figure is an example of tracking error, where the markings on the pig are incorrectly tracked as the boundary.

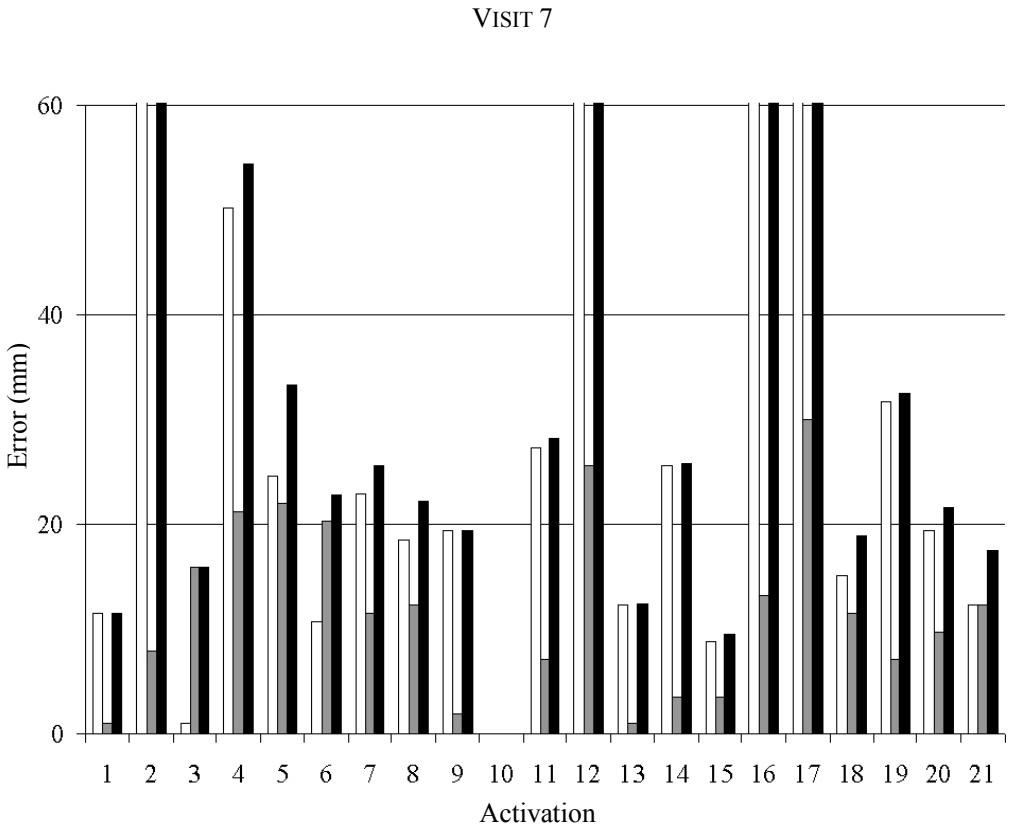
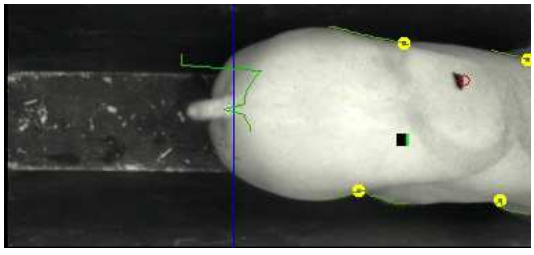


Figure 3.43 Placement error graph, visit 7. Errors between the robot sensor position as marked by the laser spot and the P2 mark on the pig's back. White = longitudinal error, grey = lateral error, black = Euclidean distance error.



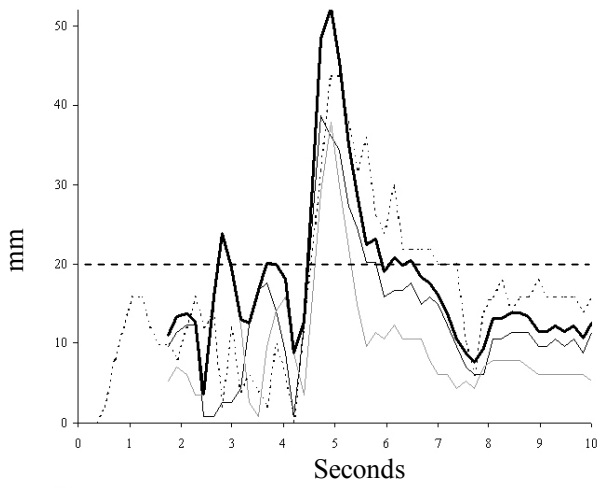
Activation image, activation 3.

Note correct kink location despite poor rump placement.

Boom image at robot arrival, activation 3.

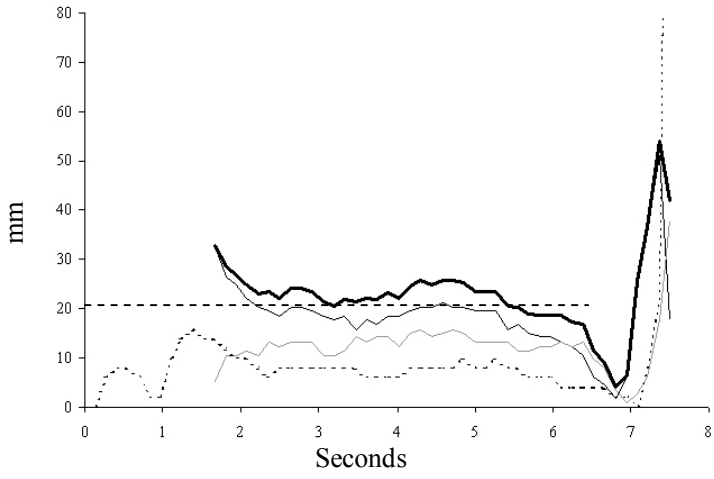
Figure 3.44 Example images. (Top) Typical activation frame output by the image analysis software at the point of robot activation, and (bottom) a boom camera frame of the resulting position of the laser when the robot is in position for activation 3.

GRAPHS SHOWING P2 POSITION ERRORS AND RUMP MOVEMENTS FOR SOME ACTIVATIONS



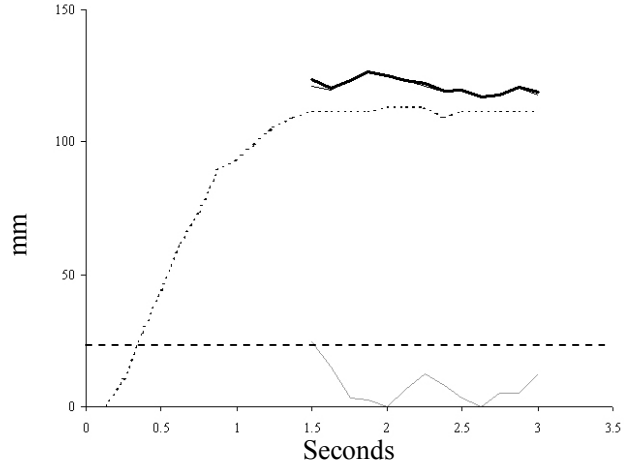
Activation 3

A restless pig can still produce an acceptable error level.



Activation 8

The pig has little motion for almost seven seconds, then extreme motion causes robot to reset



Activation 2

Pig moves as soon as robot activates. This may be caused by the pneumatic hiss of the robot on activation. Normally the pigs are not affected by this, but sometimes it startles them.

Figure 3.45 Graphs depicting a typical activation event and some atypical scenarios during visit 7. The bold line is the Euclidean distance error between the laser and the marked P2 position. The thin grey line is the lateral laser-P2 error and the thin black line is the longitudinal laser-P2 error. The dotted line represents the longitudinal movement of the pig's rump throughout the sequence. The error lines are not plotted until the robot has reached its final destination. The dashed horizontal line represents acceptable error.

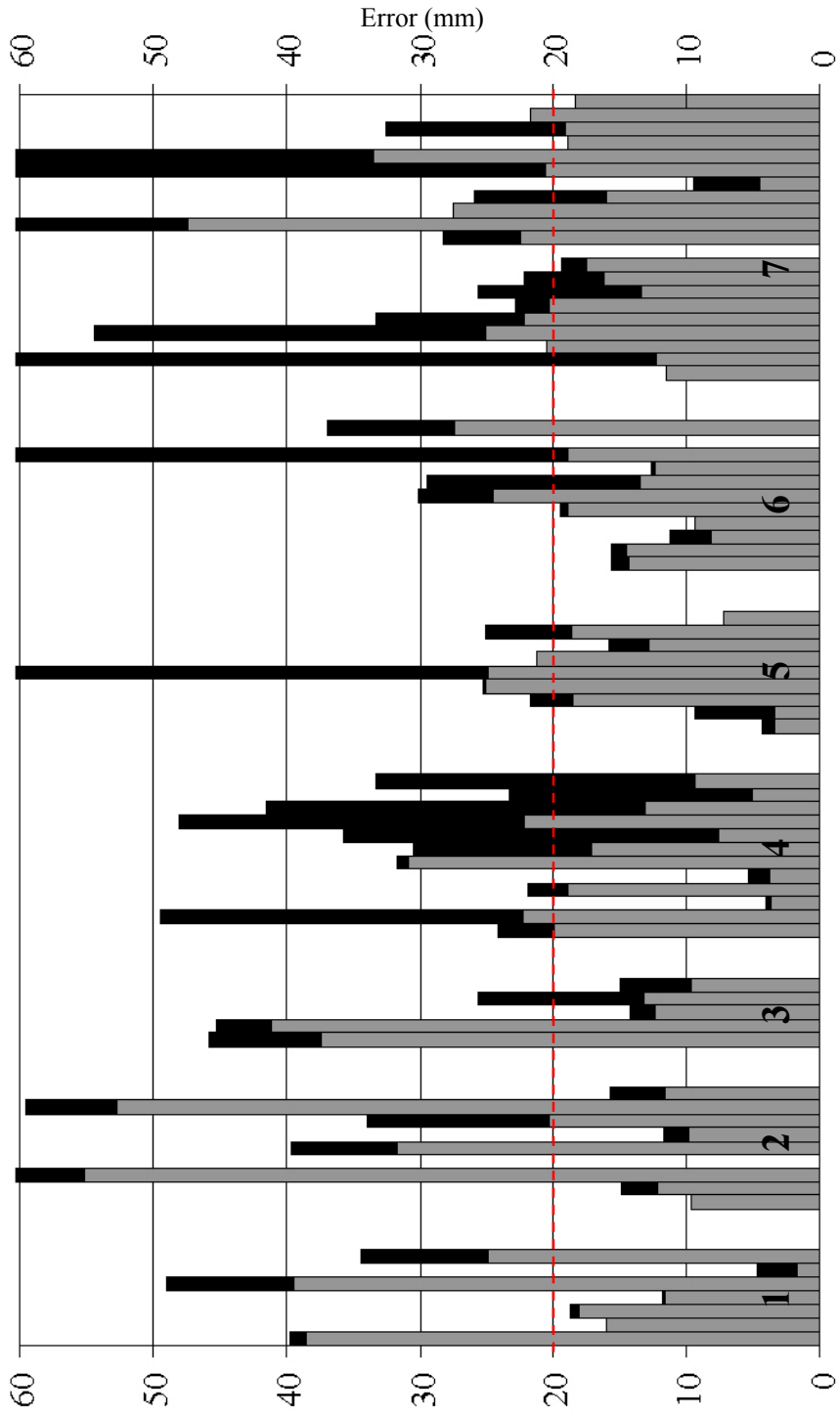


Figure 3.46 Graph to show Euclidean distance errors at sensor contact across different activations, grouped by 7 pig feeding sessions. The bars show total error (black) and adjusted error (grey) Adjusted errors refer to errors where the movement of the rump has been taken into account, and removed from the longitudinal error term. Dotted line indicates target accuracy of 20mm

Figure 3.46 shows the positional errors over the contact phases for seven sessions (two pigs are repeated). Each group represents one pig's feeding session, each bar indicates an activation by the robot. The grey bars indicate the predicated reduction in error if the longitudinal movement of the animal is taken into account. Note how accounting for the movement of the rump after the robot has been activated does improve the theoretical placement accuracy of most activations by a large amount. This indicates that, as expected, most error comes from the longitudinal movement of the pig after robot activation. For a small number of activations, the error is actually increased. In these situations the rump motion would not be responsible for the error present, and therefore accounting for the rump motion actually has a negative effect.

3.9. Discussion of results

3.9.1. Success of the system

As can be seen from the placement error graphs in section 3.8.1, 30 out of 75 placements produced a Euclidean distance error of 20mm or less. This suggests 40% of placements could be expected to be of the required accuracy. This is consistent with the 42% estimate found by preliminary analysis of these results (French et al. 2003). On four activations out of the 75, there was no available data with which to record the errors, as the marker tape had moved out of the boom camera frame. On three of these occasions, the animal backs out of the pen temporarily. This may be because the pig is startled by the noise of the pneumatics. An example of this situation can be seen in the bottom graph of Figure 3.30 on page 73. The pig sometimes steps back so far that the marker tape is no longer visible on the boom camera image and so no error can be recorded. The fourth piece of missing data is caused again by the marker tape not appearing on the camera, this time because the animal is standing in the feeding trough. However, in these situations the error would clearly be very large and so this missing data can be considered as a failed sensor application, i.e. the placement error is greater than 20mm.

It can be seen from the placement error graphs in the previous section that as a general rule, there is more longitudinal error than lateral error in the sensor placement. One reason for this is because the pig is constrained laterally by the sides of the feeder, but is free to move in and out longitudinally. Notable exceptions occur where the overall error is low, for example during visits 1 (activations 2,3,4 in Figure 3.25), 5 (activations 1,6,7,9 in Figure 3.37) and 6 (activations 1,2,3,4,5,8 in Figure 3.40). During these times, the pig is quite motionless longitudinally – it is probably happy eating – and so even though the lateral error is also small it is larger than the very small longitudinal error. The key thing to note is that when large overall errors are present, the major component of the error is the in-out longitudinal motion of the animal.

On the image analysis side, the average P2 prediction error is about 15mm. At least one part of this error stems from inaccurately located kink points. Some pigs present an outline that makes it hard for kinks to be located. Pig number 3, which visited the feeder twice, in visits 5 and 7, had a body such that the front left kink was hard to locate because only a subtle kink was present. This can be seen in the Activation images in Figure 3.38 and Figure 3.44. Hence this pig has particularly high kink location errors (visits 5 and 7, Figure 3.24).

If a pig is expected to visit the feeder for at least five minutes every day (Frost et al. 2000), and an activation occurs about every 30 seconds that a pig is in the feeder, it can be expected that there would be about 10 activations per pig per day. With 40% of activations being of sufficient accuracy, this would lead to 4 measurements of backfat per animal per day of sufficient or greater accuracy than a skilled human could achieve. However, casual observation during this work suggests pigs would spend more like 30 minutes each per day in the feeder, possibly as much as 50 minutes (Tillett et al. 2002). With 50 minutes per day in the feeder, there would be about 100 activations leading to 40 sufficiently accurate measurements per animal per day. As P2 is at a minimum in backfat thickness, of these measurements the true backfat could be taken as a minimum of these results (Tillett et al. 2002). Therefore a reasonably accurate measure of backfat could easily be expected at least once per day. As at present stockmen are likely to only

measure backfat about once per week, this is a great improvement. It is also of sufficient frequency to allow the nutritional make-up and allowance of food to be determined on a day-to-day basis.

There is a question of how to detect which are the accurate measurements and which are not. Locally, the P2 point is a minimum of backfat thickness, but globally this may not be the case. Therefore, for a fully implemented system it is necessary to detect when a measurement is a long way away from P2. One way of doing this is using the rump tracker to provide an estimate for longitudinal movement. As most of the error is longitudinal (as can be seen from the longitudinal and lateral component errors in the placement graphs of the previous section), having an estimate of this kind of error would allow the system to record backfat measurements accompanied by large rump motions as being unreliable.

3.9.2. Improvements and future work

As most of the movement and error is longitudinal, accounting for the rump motion and therefore providing a form of active tracking for the robot should produce a sizeable increase in accuracy (see Figure 3.46).

Being able to set the model offsets using something like RFID tags to be able to identify individual animals would allow the system to automatically adjust its parameters for each animal. This would also allow other data to be stored for each animal, e.g. its weight if such software was running (Schofield and Marchant 1990; Marchant et al. 1999), and this in turn could be used to provide an estimate for the height of the pig. Knowing the height is important for calculating the pixel to millimetre relationship on the pig's back, and obviously this changes as the animals grow. Such a system would also allow a conformation record to be automatically kept for each pig, providing, for example, daily weight and backfat measurements which the stockman (or future expert system) could use to tailor the animals diets or detect the onset of illness.

The idea of using the minimum of repeated backfat measures to provide an estimate for the backfat depth at the P2 point relies on the fact that P2 is at a minimum of backfat depth. This is a *local* minimum, but previous work suggests the P2 point is a minimum within an area of at least 50mm from the true position (Tillett et al. 2002), giving quite a large margin for error, and assuming enough repetitions are available it would allow estimates to get close to the true value. Clearly, if this system were implemented for real, there would be no way of automatically telling where the *true* P2 point was. If the robot positions the sensor a long way from the correct point, then the local minimum rule would not hold, and erratic values for P2 backfat thickness may be recorded. This problem could be overcome by either developing an error detection mechanism that could detect when the predicted P2 point or final robot position was unreasonable (e.g. after large rump movements), or by assuming that enough placements will be within the valid local minimum area that a mean value (or other measure, dependant on the distribution of points) would be a sensible estimate of P2 backfat thickness.

It is possible that the backfat distribution around the P2 point is consistent enough across animals that its thickness at particular locations could be modelled. This may allow estimated locations and readings using the robotic system to be fitted to the model. In turn, this would allow multiple measurements to provide the ability to fit the modelled thickness map to a particular pig's back. This may perhaps allow a much more accurate location of the true P2 point in the presence of noisy data. Additionally, if during a set of backfat readings the true minimum is in fact *not* located, the model could be used to identify the theoretical position of the minimum thickness of backfat using the available data.

Some practical experimental considerations were raised by this work. First, the boom mounted camera was on rare occasions unable to capture the laser point and marker point in the frame. This was the cause of some missing data in some of the graphs (e.g. sequences 10 and 12, Figure 3.40). This could be rectified in the future by using a wider angle lens.

The image analysis system itself did prove generally reliable, although in testing the rump tracker did occasionally get caught on clutter introduced into the pen by the animals. The best way to remedy this was to keep the area around the entrance to the robot clear of bedding material, which in turn allowed the floor of the feeder to be kept reasonably clean. A future way of detecting when the rump tracker was caught on background clutter is to look for occasions when the rump tracker becomes completely motionless – this rarely if ever occurs when on the animals. Testing for such an event would allow the system to know that clutter was present, and the image analysis could try reinitializing itself or alerting an operator to clean out the feeder.

Another issue was raised about how accustomed the animals must be for them to accept the robot. The animals used in these experiments were raised with the robot in the animal house, and they had access to it at all times. However, some early tests with unaccustomed adult pigs suggested that the noise of the pneumatic system may cause unhabituated animals to be startled when the robot activates. This may cause them to back out the feeder completely, or at least be more jumpy as they are feeding, causing any prediction of the P2 to only be valid for a short period of time. A remedy for this would be to silence the pneumatic system, or raise all animals with the robot present.

The environment itself proved challenging, especially the levels of dust. Although this did not affect the equipment used in this work, any long term installation would need to be thoroughly dust-proofed.

Chapter 4: Monitoring Multiple Animals by Visual Tracking: Video Tracking of Ducks in an Outside Arena

4.1. An extension of single animal monitoring

The previous chapter presented a novel method of monitoring individual animals. This consisted of a computer vision system controlling a sensor placement robot. It was shown that this system provided enough readings per day of human accuracy or better to improve greatly on the current system of manual measurement by skilled operatives. This previous work showed that useful information could be gathered from automated monitoring systems tracking individual animals. An image analysis monitoring system for *multiple* animals will now be presented. This and the following chapters detail the development of a system which can support remote visual sensing, i.e. non-contact and at a distance from the subjects, by tracking multiple animals.

4.2. Introduction

4.2.1. Aim

It is the aim of this chapter to test existing tracking techniques in a novel application area. The practical goal is to track a group of animals in an outside environment as they go about their activities, with a view to further monitoring applications. The techniques will be tested on a group of ducks in an outside enclosure. To support future monitoring applications, all animals in a scene

should be able to be located at each timestep. Therefore, this work requires that multiple targets be tracked reliably. The targets here are similar looking, often in close proximity and they sometimes interact with each other. This chapter will examine how suitable existing methods are for such tracking of groups of animals. The ability to extend to an actual monitoring system would require robust tracking over long periods of time, so this work is concerned mainly with the reliable and robust tracking of the animals. The ability of social motion information to aid tracking will be investigated in the next chapter where a new tracking method is developed.

4.2.2. Multiple target versus single target monitoring

The monitoring of one animal as presented in the previous chapter does not provide as many possibilities nor as many challenges for tracking algorithms as monitoring multiple animals does. With only one target, the main distractions or occlusions the tracking system has to cope with are generated by background clutter and self-occlusion. While this may in some situations be a significant challenge in itself, often the problems generated by the background can be minimised by careful planning. In the case of the pig monitoring system, the robot feeding rig was coloured black to contrast against the pale pigs, and kept as clean as possible to minimise the amount of ambiguity between foreground and background. Self-occlusion was not a problem as no parts of the pigs' bodies were able to occlude the features being tracked. Although the robot arm did occlude some of the features being tracked, this problem could be effectively worked around because the occlusion occurred at a predictable place and time. When multiple interacting animals are tracked it is no longer just the background that can be the cause of cluttered measurements, but the animals themselves. This is especially true if the animals are similar in colour, as is the case with the ducks. With animals, this situation is common. Groups of farmed animals typically all look very similar (e.g. white sheep, white ducks, lab rodents etc.) Even animals that may start out with unique features may over time look similar after becoming muddied in an agricultural environment. Additionally, depending on the camera angle, the animals may occlude each other to a varying extent when they interact in close proximity. Multiple single target trackers easily get distracted in such

interactions, as will be shown. Therefore, the problem of tracking multiple similar targets is one that needs to be addressed in this domain. This problem is not unique to animals; similar targets in close proximity occur in many real world situations as well. With people, clothing often allows targets to be uniquely identified, but some camera angles, such as top-down views, cannot differentiate so well between targets. Some non-vision sensing systems such as sonar and radar often are unable to differentiate between similar targets as well as vision systems.

Being able to track multiple targets is very important for automated surveillance systems. By tracking multiple targets as opposed to individuals, or rather by tracking individuals but being aware of where other individuals are and how they are moving, much more information is available for further processing. For example, someone looking for suspicious behaviour might look for groups of people moving about a scene in a suspicious way - a group here may be quite separated spatially but acting in a common fashion. An animal behaviourist might be interested in how animals are moving as a group in response to certain stimuli (Henderson 1999). They may also be interested in where individuals are within a group, as this may indicate a trade off between certain driving forces, e.g. the safety of the centre of the group versus the food availability of the periphery (Rayor and Uetz 1990). Team sports analysis is also a growing field (Needham and Boyle 2001), and what makes certain teamplay strong or weak can be of interest to coaches. As well as classifying the behaviour of the whole group, individuals within the group can be classified depending on how they are moving relative to the rest of the group. For example, a lame animal may move with a group of other animals, but perhaps exhibiting an oscillation on its trajectory. Other intra-group relationships might include someone being chased and their group of pursuers, or a group of thieves surrounding a victim. None of these effects can be identified by observing only individual targets; the behaviour of the whole collection of targets must be considered.

When considering multiple targets that have a degree of interaction with each other, there are interesting social effects which occur that can both hinder and

potentially aid tracking, such as how groups of targets move as a group, flock or herd. Their effect on tracking will be seen in this chapter, and Chapter 5 will examine ways of using social effects to *aid* tracking.

4.2.3. Why ducks?

It is reasonable to ask why ducks were chosen as the animals with which to develop and test the tracking algorithms. Ducks are social animals that demonstrate grouping and interactive behaviour. Because of these properties, ducks are ideal subjects. From an experimenter's perspective, ducks are more easily managed than larger animals, such as sheep. It is also possible to keep more ducks in a fixed arena size than larger livestock. The ducks used were Pekin variety, which are a consistent white colour. This challenges the tracker to distinguish similar targets which can present problems even to humans, and so is useful to investigate. Ducks have been used successfully in previous flocking experiments (Henderson 1999; Sumpter 1999; Vaughan et al. 2000). Once acclimatised to an arena, they can be filmed unobtrusively without concern about the ethics of people appearing in videos they might not want to appear in, as might be the case if 'natural' footage were acquired from a public space. Using human 'actors' would not be ideal because the subtle behaviours which would coincide with 'group motion' would not be able to be extracted; they would only exist if the actors were told to perform a certain action.

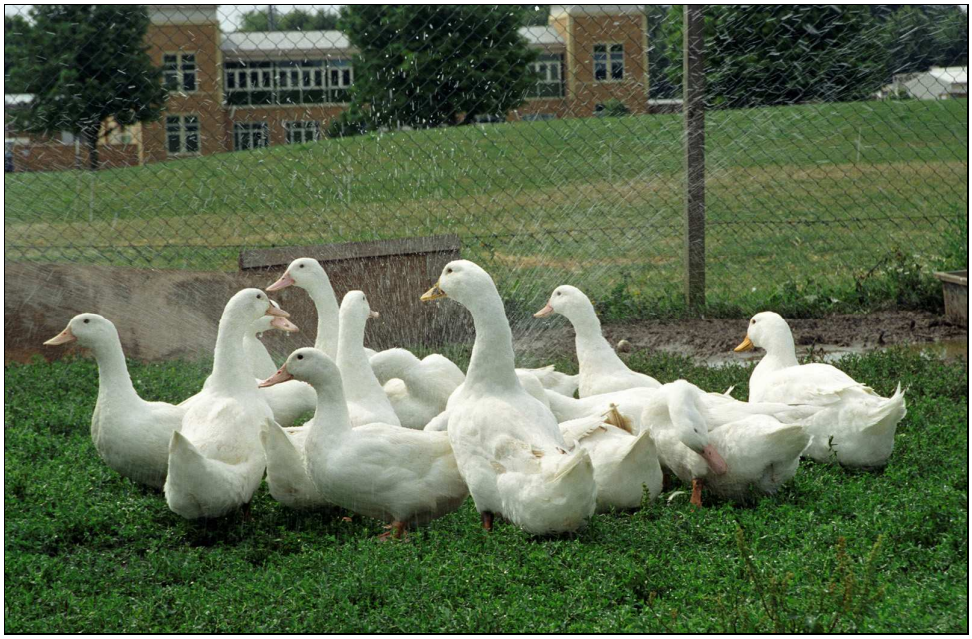


Figure 4.1 The experimental subjects; a group of Pekin ducks at the outside enclosure at Silsoe Research Institute, Beds., UK.

Being able to track the ducks reliably would allow the development of a monitoring system which could for example identify lame animals moving differently to the group, animals eating or drinking too often or too infrequently, or startled animals flocking together away from a predator, in either case providing an early warning for the stockman. Such a system would be useful because it would allow improved welfare for the animals, and hence improved quality and quantity of product and also financial reward for the farmer.

4.3. Literature review: Multiple target monitoring

4.3.1. Applications for tracking multiple targets

This work will look at the tracking of multiple ducks: however, such methods can be extended to tracking groups of people, other animals, insects or any other groups of multiple targets. Many interesting possibilities arise if such targets within a scene *could* be tracked reliably. Although the idea of a Big Brother society is currently as implausible as it is infamous, it is not hard to imagine situations where it is useful to be able to monitor the actions of many individuals that together make up a scene, without sinister Orwellian connotations. In turn, the group motions, behaviours and interactions between the targets may be of interest to many different types of people or organisations. The people interested in such monitoring may not be the obvious choices that first come to mind, such as the police forces, military, or other public order-related people. In fact, monitoring people's interactions with each other can be used in healthcare scenarios, such as looking at people's social interactions in care homes (Chen et al. 2004). In education, tracking many children at once can be used to provide an informative and (importantly) fun interactive social environment in which to learn (Stanton et al. 2001). Such interactive social learning has become popular in recent years, for example, the Kidstory project (Swedish Institute of Computer Science 2005), or Kidsroom (Intille et al. 1997) and adding group tracking elements to these provides interesting research and educational possibilities.

One typical example situation is in the team sport domain. When people play team sports they are operating in a multi-person group with various goals. Analysing this kind of behaviour allows the labelling of sporting events. For example in the world of American football, players motions have been analysed to label coordinated group play (Intille and Bobick 2001). These formations exist as a pre-defined taxonomy and so identifying them is easier then recognising the more flexible formations of less structured situations, such as animal social activity or crowds of people. The algorithm used performs well (21 out of 25 plays correctly identified), though some plays are confused. However, the rule-based descriptions of the formations are quite specific and will not generalise to

more flexible systems. For example, the system cannot identify a move that it has not yet been taught: it will try and classify as best as it can, producing false positives. Additionally, every target in the sequences forms a part of a play: everything the targets do happens for a reason. In 'real world' problems this assumption will not hold, as perhaps not everyone in the scene will have as clearly defined goals, or even no goals at all. It should also be noted that the American football work (Intille and Bobick 2001) does not feature automated video tracking; positions are manually entered at high labour cost. However, this analysis could be applied to the results of an accurate tracking system if a good enough system existed. Other systems have attempted the team sports tracking problem. Basketball players have been tracked whilst on court (Needham and Boyle 2001) using a hybrid Condensation/Kalman filter tracking system, which results in 56% of automatically produced trajectories falling within one metre of the groundtruth. This is a complicated sequence as the targets often occlude each other; however, if 44% of the tracks were too inaccurate for behavioural analysis, the analysis of the whole team would not be possible.

If targets' motions can be robustly recorded, a map of which areas of an image are visited can be built up. Previous work on individual or small numbers of targets has identified routes across public areas such as carparks (Makris and Ellis 2002). Typically in existing work on this subject, although the statistics are formed from many people's paths through the scene, the people themselves are spread out temporally; the tracking would begin to fail where many targets are present at once. Being able to perform such analysis in *busy* public areas would allow inferences to be made about the environment in which the crowds are moving, for example areas that are popular or avoided in shopping malls could be used to aid the design of future centres. Some work exists which uses texture analysis to estimate the density of a crowd (Marana et al. 1998; Chow and Cho 2002), but being able to track the individuals that make up such crowds is an important step towards being able to make more informed judgments about the current and future *behaviours* of the crowd. Identifying such behaviours might consist of crude judgements based on the velocity of the members of the crowd, to detect when they are panicking or rioting, for example. The drawback of tracking individuals

is that at the moment technology limits the number of people that can be tracked: macro-scale movements are more feasibly tracked than individual level motions when the number of targets is large. Therefore, it is necessary to consider smaller groups as opposed to larger crowds with current technology, but it is a first step.

Monitoring applied specifically to multiple animals has clear advantages mentioned previously in the context of single animal monitoring: improved animal welfare, increased profits for the stockman etc. Most monitoring takes the form of measuring a characteristic of an animal, such as its temperature, weight or conformity. However, it is also possible to monitor using a visual tracking system alone. For example, the distances animals travel can be indicative of their state of wellbeing (Brandl 2005). Also, how animals react to each other can show something about their feelings: feather pecking in turkeys (Savory 1995) is one example of how animals' attitudes towards each other can be directly observed. The *group behaviour* of multiple individual animals can also lend insight into their well being. For example, pig group behaviour has been used as an indicator as to whether the temperature of the environment is correct (Wouters et al. 1990). Although Wouters' system is very simple (using thresholding to identify sleeping areas of pigs), it demonstrates that it is possible to automatically control environment conditions using image analysis to extract group information from scenes. Also, group behaviour can be used to identify the presence of a predator, as grouping characteristics change in the presence of a threat (Henderson 1999). Being able to track individual animals for work used to analyse group behaviour provides much more information than just looking at the movement of the group as a whole; such non-individual group tracking has been popular previously (Sumpter et al. 1997) presumably because it is faster, more reliable and provides sufficient information for the task at hand.

4.3.2. Tracking: a review of relevant techniques and theory

In order to monitor groups of animals, they must first be tracked reliably. This means reliably locating targets in successive frames and maintaining their identities throughout time, to enable their position and motion to be quantified.

Existing work on various tracking methodologies and algorithms will now be described, with a view to selecting suitable algorithms for this work.

The crudest way of being able to identify which targets in the successive frame correspond to the targets in the current frame is simply by seeing which target in the new frame is closest to the target's old position in the previous frame. Typically a local search window will be placed in the new frame, centred on the old target position. If only one target is found in this window, then it is assumed to be the required one. If more than one is present, there may be a heuristic for choosing between them (e.g. based on an appearance model, or a distance metric), or one of the new target candidates may simply be randomly allocated the identity of the old target. This method is only going to succeed where there are a small number of targets, unlikely to be located close to one another, and where the velocities of the targets are small enough to keep the targets within the search window between frames.

A simple extension to this technique which makes this approach more powerful is to take account of the target's velocities, and move the search window accordingly. Therefore, a target with a fast motion will have it's search window moved further ahead in the next frame than a target which is moving more slowly. A further improvement is to alter the size of the search window depending on how confident the prediction of the target's new location is. If the tracker is confident of the target's predicted location (e.g. if all the previous recent predictions have been accurate) then the search window can be made smaller than if the tracker is not confident of the prediction. A smaller search window provides less opportunity for a tracker to pick up on background clutter on incorrect targets.

Figure 4.2 illustrates this principle.

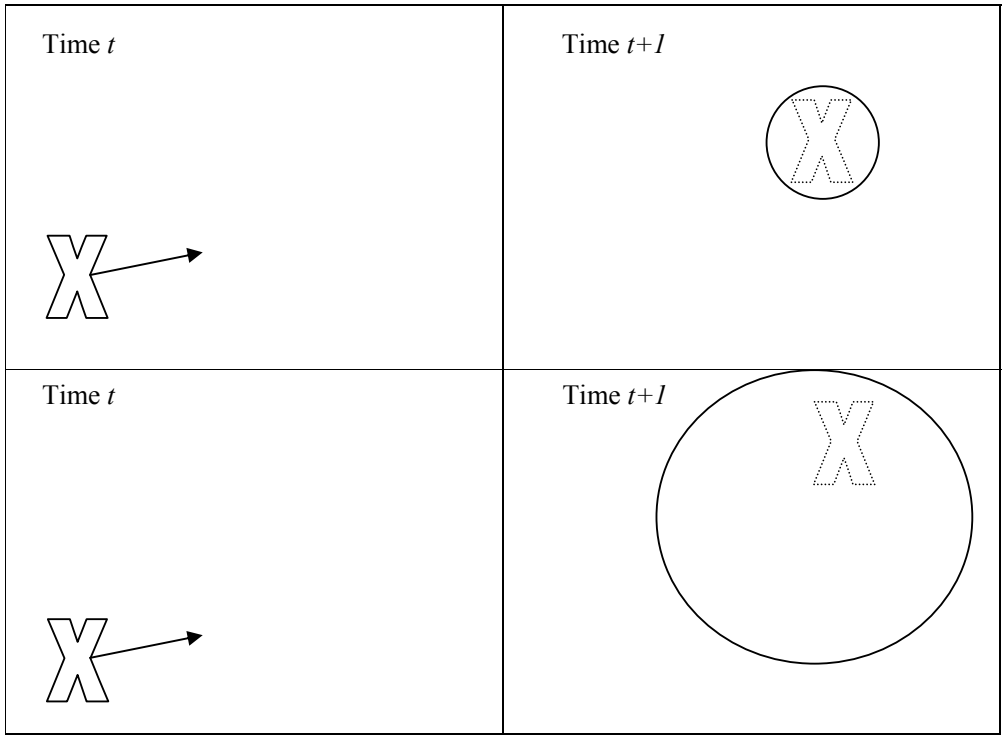


Figure 4.2 Examples of tracking a target ('x') using a circular search window placed using the target's velocity to estimate a new position and with a radius based on the confidence of the target being in that location. The top frames represent a tracker confident of the target's location (small search region) and the bottom frames show a less confident tracker (larger search region). In the bottom t+1 frame the target is misplaced due to noise, but is still captured by the larger search window. Note that in both situations a search window placed at the target's old location (i.e. discounting velocity) would fail to locate the target.

With a high confidence level, the window could be shrunk to both speed up the search for the target and to lower the chance of latching on to the wrong target. Integrating these two fundamental ideas into a tracker leads to the Kalman Filter.

KALMAN FILTER

One of the most widely used and powerful algorithms used in tracking is the Kalman Filter (Kalman 1960; Welch and Bishop 2001). It has been in existence for over 40 years, and has recently seen resurgence with its use in computer vision tracking problems. It is a linear predictor-corrector estimation algorithm, meaning that a prediction is made and then refined or corrected based on a measurement. It is both simple and robust, and optimal in the sense of minimizing the covariance of the estimated error, and because it incorporates *all* available data. It is robust

in that the filter demonstrably works very well in a variety of situations despite the conditions for optimality not being fulfilled. It is also recursive, meaning that it is not necessary to store and reprocess all the preceding data every time a new measurement is taken. The basis of the algorithm lies with Bayes' rule, as is the case with many probabilistic trackers. Bayes' formula allows the calculation of probabilities using easier-to-implement causal (versus diagnostic) reasoning.

$$p(\mathbf{x} \mid \mathbf{z}) = \eta p(\mathbf{z} \mid \mathbf{x}) p(\mathbf{x}) \tag{4}$$

where $p(\ x \mid z\ )$ is the posterior distribution, $p(\ z \mid x\ )$ is some reinforcing measurement, $p(\ x\ )$ is the prior distribution, and η is a constant.
Thus it is possible to determine the diagnostic left-hand-side by evaluating the causal right-hand-side. This is useful in tracking because it allows probabilistic rules to be inferred from observed data. The Kalman filter makes use of this rule by combining a prior estimate and a measurement to compute the posterior state estimate.

There are two major steps in Kalman filtering: predicting from the current state ahead in time, and adjusting this prediction using an actual measurement. Both the current state and the error covariance estimates are projected forward in time. This allows the filter to estimate a location for the target, and a confidence of this location (the *a priori* estimate). This confidence can be used to determine the size of the search area in a tracking algorithm. Feedback about the quality of this estimate comes in the form of a measurement, which can correct the estimates of location and error to provide an *a posteriori* estimate. The motion process is modelled using a linear stochastic difference equation. The error covariance is projected forward at the same time as the motion process is applied, and updated when a measurement has been made.

For visual tracking purposes, the algorithm works by predicting the location of the target in the next frame and quantifying the variance of the estimate. This allows a search window to be located in the new frame at a position based on the target's previous motion, and with a size proportional to the noise in the estimate. There

are two sources of this noise: process noise and measurement noise. The location is then refined using a measurement of the target from the image, and new estimates of error and motion calculated. This algorithm proves to be very good at tracking individual targets with Gaussian measurement noise, moving in a linear fashion. Such a Gaussian distribution looks like:

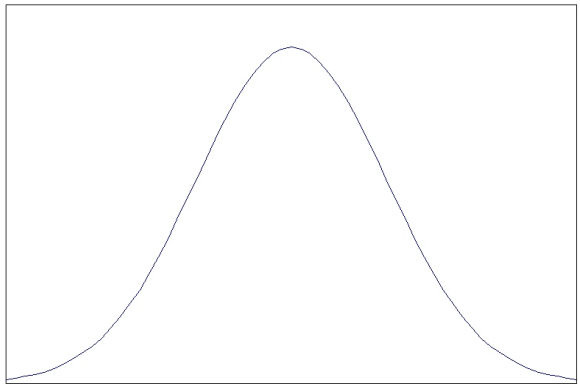


Figure 4.3 Gaussian normal distribution curve.

In fact, the tracker can be extended to allow for non-linear dynamics. The Extended Kalman filter is a way of applying the Kalman filter to non-linear processes by locally linearizing the system.

Although the Kalman filter is optimal and successful in some situations, there are occasions where it cannot be used. The algorithm makes three fundamental assumptions for optimality to hold. First, the system must be considered linear. Although this is often not the case, this assumption can be justified because most processes can be approximated to be linear over short distances. It is sometimes also possible to move around this assumption with the Extended Kalman filter. The second assumption is that the noise frequency values can be considered 'white'. Whiteness implies that the noise cannot be correlated in time and that the noise has equal power at all frequencies. In fact, this situation is impossible in reality as it would consist of the noise having infinite power. However, because all physical systems have a bandpass of useful input, the white noise approximation can be applied between these bands. White noise is a simpler model than actually modelling the bandpass noise, and so the white noise model is used. Finally, all the noise amplitudes are assumed to be Gaussian. It is this final

point that prevents the Kalman Filter from being used when multiple targets or significant background clutter are present. Measurements taken in the presence of target distracters, i.e. similar background clutter or other similar-looking targets is likely to produce a measurement probability distribution which is non-Gaussian, as illustrated in Figure 4.4.

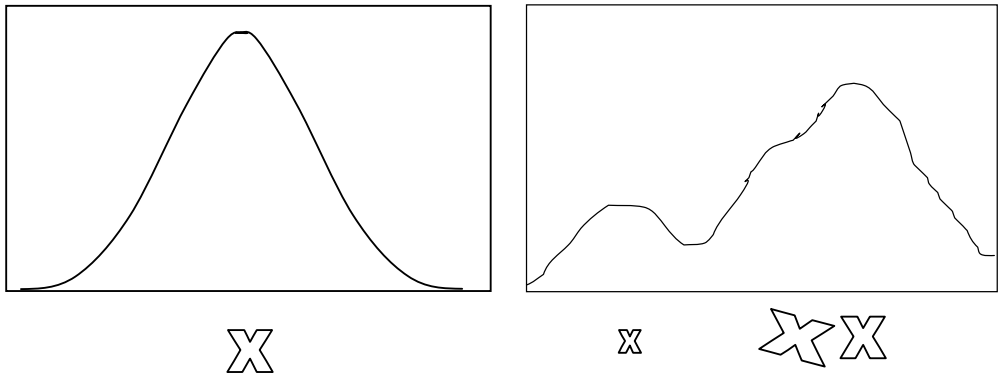


Figure 4.4 Measurement probability distributions for one target in no clutter (left) and a target in similar clutter (right). Note that the left-hand distribution is normally distributed, and the right hand distribution is not. Hence the left-hand distribution can be represented with a Gaussian distribution with the appropriate parameters. The right-hand distribution, however, has no such simple, closed form representation.

A problem inherent in multiple target tracking is confusing clutter, multiple 'correct' measurements (i.e. many targets) and hence a potentially non-Gaussian probability distribution function. A Kalman filter would not be able to represent this situation effectively.

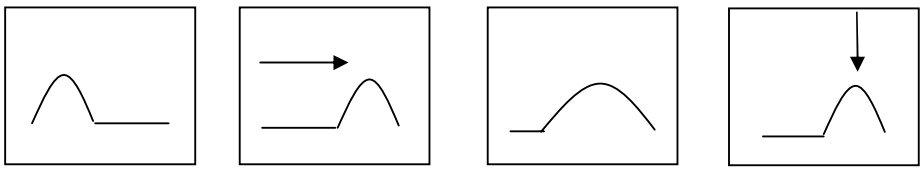
Kalman filtering has, despite this, been used as a component in multiple target tracking methods, for example the Multiple Hypothesis Tracker (Reid 1979). Here probabilities are used to assign measurements to targets, and then a Kalman filter is used to drive the state estimation from such hypotheses. This method has a number of drawbacks though, not least of which is that it expects an inflow of new targets into the surveillance region, and can in fact initiate new target tracking from one measurement. This is bad news for any scenes with background clutter, or for any scenarios where the number of targets is fixed. The Joint Probabilistic Data Association Filter or JPDAF (Bar-Shalom et al. 1980) attempts to eliminate some of the problems of the Multiple Hypothesis Tracker.

This algorithm handles the association of an arbitrary number of measurements at a given time to an arbitrary number of established targets, i.e. no new targets are accounted for. However, this algorithm itself has drawbacks, including its inability to handle occlusions well. As image likelihoods are evaluated independently, tracking can break down when targets become close to one another or overlap, and no mechanism is given to overcome this problem.

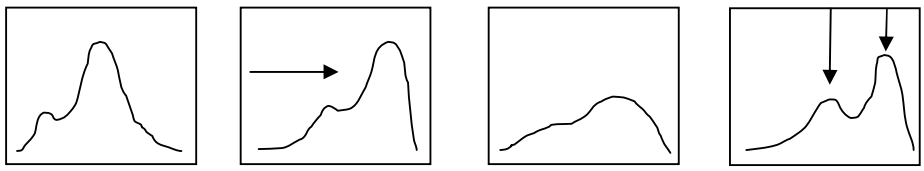
Therefore a more powerful approach is required, which can handle target tracking through heavy background clutter, with the potential to be extended to multiple target situations. Such a solution is provided by the Condensation algorithm.

CONDENSATION

Condensation (Isard and Blake 1998b) was developed to cope with non-Gaussian distributions and hence the multiple hypotheses present when tracking with dense visual clutter or multiple targets (see Figure 4.5), exactly the situation that Kalman filtering cannot cope with and precisely the situation we are presented with when addressing the problem of tracking multiple similar targets.



Kalman density propagation. Left to right: the process of deterministic drift, followed by stochastic diffusion and lastly the reactive effect of an example measurement.



Condensation density propagation. Left to right: the process of deterministic drift, followed by stochastic diffusion and lastly the reactive effect of 2 example measurements.

Figure 4.5 Density propagation: Kalman filtering vs Condensation.

Condensation uses factored sampling, a method of stochastically representing a probability distribution. The probability density function is approximated by a discrete set of 'samples' or 'particles', $\{\mathbf{X}_t\}$. Each sample, X_t, contains a

complete representation of the parameter vector of the target at time t and a weight, π_t. Samples are propagated forward based on how well they match the measurement; this match is quantified by weighting the samples appropriately by evaluating an observation density, typically but not necessarily, a Gaussian. Every iteration, samples are selected with a probability proportional to their weight. This allows likely hypotheses to be propagated forward in time.

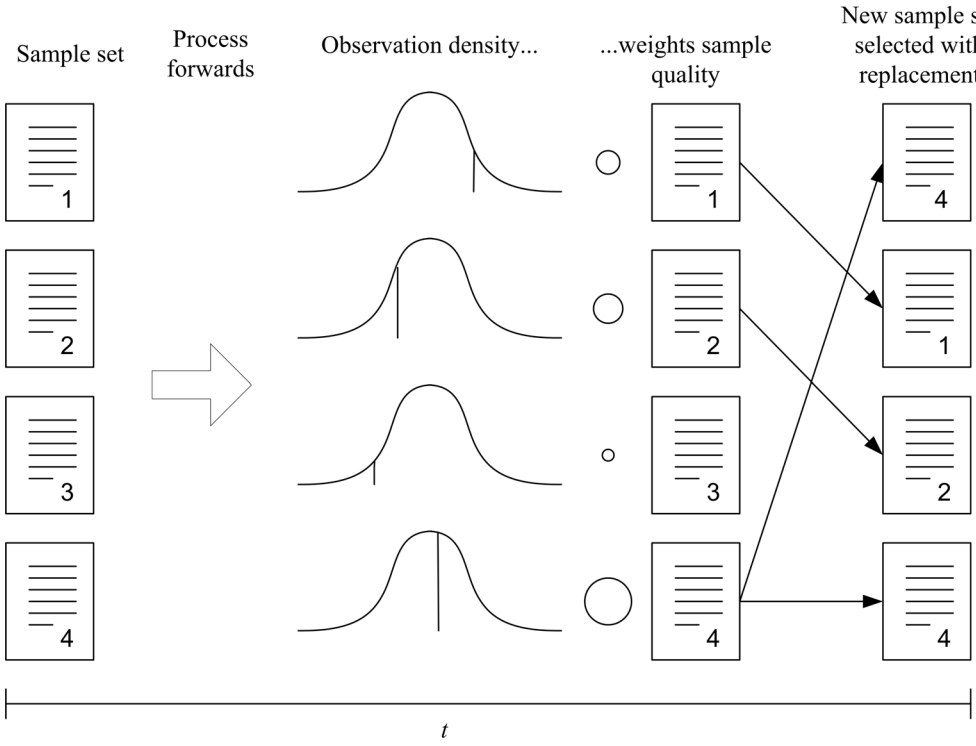


Figure 4.6 One time step in the condensation algorithm. Each box represents one sample, $\boldsymbol{X_t}$, and each box holds a complete description of a target's state. The size of the circles represent the weights of the particles.

There are several extensions to Condensation already in the literature. These will be summarised in turn, as each provides a potentially useful enhancement to the basic framework. The ability of each to aid tracking will be discussed.

EXTENSIONS TO CONDENSATION

A target moving in more than one distinct way can cause problems for traditional single-motion-model trackers. A method of coping with more than one model of motion is introduced using mixed state Condensation tracking (Isard and Blake

1998a). This work introduces an extra discrete variable to the particle state that flags which motion model to use. The new extended state now looks like:

$$\mathbf{X} = (\mathbf{x}, y), y \in \{1, \ldots, N_S\} \quad (\mathbf{x} \text{ is the parameter vector, } y \text{ is a state label})$$

A matrix of model-state transition probabilities is supplied, and used to process the discrete state label y forward in time. Using this model, transitions between states occur automatically, as each state transition with non-zero probability contributes samples to the distribution. As one model predicts more accurately, more samples from this model will be propagated and this model will dominate. Being in a different state basically means using a different motion model to process the samples forward in time. This mixed state, model-switching approach is used in the literature to successfully track a bouncing ball where traditional single-state Condensation fails. It is also used to track a hand drawing a picture, and to assign one of three states to the hand (line drawing, stationary, scribbling). The tracking is successful, at the cost of running more slowly than single-state Condensation, due to more samples being needed as more models are used. Understanding the complexity of learning a mixed-state motion model is highlighted for possible future research. Magee and Boyle (Magee and Boyle 2002) use Hidden Markov Models to assign discrete states (lame and healthy) to their version of 're-sampling condensation'. They successfully use condensation to track a walking gait and assign a discrete label to the motion, in a similar manner to mixed state Condensation.

One feature of Condensation is that the discrete nature of the sample set means the samples cluster around areas of high probability and large areas of the state space contain no samples at all. This allows high dimensional state spaces to be efficiently represented. It also means, however, that to capture sudden unexpected changes in the motion or shape of the target, the noise level in the motion model must be set at a high level. To prevent these new expanded clusters of samples from being too sparsely populated, the total number of samples, N, must be significantly increased. This causes the system to run more slowly. ICondensation (Isard and Blake 1998c), or Condensation with Importance

sampling, has been developed to help counter these issues. Auxiliary knowledge allows an importance function, $g(\mathbf{X})$, to be constructed which describes the areas of state space that contain most information about the posterior. Samples are then concentrated in these areas of $g(\mathbf{X})$ rather than sampling from the prior, $p(\mathbf{X})$. The overall goal is to avoid samples with very low weights, as these provide only a negligible contribution to the posterior. If the samples all have about the same weight ($\sim 1/N$) then the estimated effective number of samples $\approx N$, as can be seen from Doucet's (Doucet 1998) estimated effective sample size formula,

$$\hat{N} = \left(\sum_{i=1}^{N} \pi_i^2 \right)^{-1} \tag{5}$$

However, if only one of the N weights is significant, the effective sample size tends to one. So the aim is to reduce the number of 'useless' samples as these do not contribute to the effective sample size.

In their example (Isard and Blake 1998c), the importance function is derived from a measure of skin-colour in the image. New samples are generated one of three ways: from an initialisation prior (to re-initialise lost tracking automatically), from the standard Condensation algorithm, or finally using the importance sampling method. As $g(\mathbf{X})$ is drawn from a simple 2d blob tracker, this is only used to set the translation components of the new state vector using Importance sampling. Additional parameters (e.g. deformations) are drawn by sampling using traditional Condensation methods. The new 'importance sampled' state vectors look like $\mathbf{s}^{(n)} = \mathbf{s}^{(n)TRANS} \oplus \mathbf{s}^{(n)DEFORM}$, where $s^{(n)}$ is a sample processed by either the TRANS component from $g(\mathbf{X})$, or the DEFORM component which is sampled from the prior distribution as per standard Condensation.

What this all means in real terms is that when generating samples, some will be processed using traditional condensation, some will be positioned according to an initialisation prior (e.g. where the target is likely to appear in the scene – around a doorway for example) and some will be placed using some external probability function (in this case, near areas of skin colour). The results suggest the tracker to be very robust over clutter for all the users who have tested it, and it runs in real

time (where $N = 400$). The main advantage of ICondensation is the way in which it combines powerful but slow high level techniques (such as contour tracking where the shape of an object is tracked) with fast, low level ones such as blob tracking.

'Partitioned sampling' is another, similar way of tackling the problem of improving particle set representation efficiency (MacCormick and Blake 2000). "Partitioning" refers to decomposing the dynamics into two stages (e.g. x and y directions), applied in sequence with weighted resampling in-between. The stages of one time step of partitioned sampling are:

1. Apply first partition of the dynamics to all particles (e.g. x direction)
2. Weighted resampling with respect to an importance function
3. Apply second partition of the dynamics to all particles (e.g. y direction)
4. Weight particles using the likelihood

Weighted resampling has a similar effect to importance sampling, though it is faster – to the order of $O(n)$ versus $O(n^2)$. Partitioned sampling works effectively: results indicate that partitioned sampling can produce successful tracking when unpartitioned sampling fails, and with only a quarter as many particles.

One characteristic of Condensation, and one particularly relevant to multiple target tracking, is that Condensation quickly latches onto the "best" target where multiple similar targets are present. This is generally considered an advantage – the tracker will stay locked on an ideal target and will not be confused by similar (but not identical) clutter. However, in the situation of tracking groups of ducks, initialising a tracker on each duck may lead to the trackers jumping onto 'ideal' ducks when they are near by. The propagation of samples is so effective that after only a few frames of switching targets, the tracker has forgotten about the previous hypothesis and recovery is impossible. A related problem is where one object occludes another and the trackers both lock onto the foreground target. One method of preventing this happening is using an observation density that exhibits a *probabilistic exclusion principle* (MacCormick and Blake 2000). This is a way of preventing the presence of two targets to be inferred from the

measurements of only one. The example presented is one of tracking contours around two similar shapes where one occludes the other. Measurements are taken using a 1d feature detector along some normals on the curve to detect edges. A generative model is developed whereby the number of detections on each line is predicted where there is no target (i.e. background clutter), one target, and two targets present. The emphasis of this work lies with using these normal measurement lines, and transferring the ideas to tracking without using this specific method of measurement is not assessed for viability, other than to say it is hoped that it can be generalised successfully. The underlying concept is that any single measurement should reinforce multiple hypotheses coherently. It is not clear how well this technique will extend to tracking many targets ("implementation difficulties" are hinted at in the conclusion) or with using the colour measurement process to be used in this work. However, it should be noted that following more development this might prove a useful technique with which to track multiple targets in the future.

Some of the most promising work on tracking multiple similar targets is by Khan *et al* (Khan et al. 2003; Khan et al. 2004). Their basic hypothesis is that multiple targets in close proximity can and do influence each other's behaviour. In the first of these papers, a Markov random field (MRF) motion model is used to model the interactions between targets. The tracking of ants is very successful, however because the *joint* state space of all targets is required, the particle filter suffers from exponential complexity in the number of targets (Khan et al. 2003). The second of these papers replaces the traditional sampling step of particle filters with a Markov chain Monte Carlo (MCMC) sampling step. This allows a more efficient representation of the joint state space, and with the MRF interaction function produces good quality tracking of multiple insects.

4.4. Tracking of ducks: the algorithms that will be tested

Following on from the previous summary of existing work, this section will describe the reasons for and against using particular algorithms to track multiple ducks.

The simplest methods, such as the nearest neighbour methods already described will clearly be unsuitable because of the similarity and close proximity of the targets. Not accounting for the targets' motions would make tracking interacting similar targets something of a lottery. The Kalman filter's motion process would help to track the ducks by using their motion information, and the error covariance could minimize the search window, providing less opportunities for incorrect targets to be found within the search window. However, with the multiple similar targets present, and an inability to represent a multi-modal probability distribution, the use of a Kaman filter was discounted. The Multiple Hypothesis Tracker was felt unsuitable as it expects new targets to appear in the scene, and the JPDAF's inability to handle occlusion and close proximity of targets was felt unsuitable for this work, and so these were not implemented.

Many extensions to condensation exist; some relevant ones were presented in the previous section. As this thesis will not employ a high level, high dimensional contour approach to tracking the ducks, ICondensation is not advantageous, as a bridge is not needed from any slower, higher-level tracking to the faster, lower-level tracking methodologies. The probabilistic exclusion principle extension was not implemented as it was unclear how well it would extend to many targets, and because it was developed in its raw form for two wireframe targets using an edge-based measurement model. Extending to large groups of opaque targets using a colour measurement model was thought to be moving outside the scope of the method. Partitioned sampling has been demonstrated to reduce the number of samples needed to represent the set, but it was thought that it would become confused in the presence of multiple similar targets as the importance function may latch on to measurements from nearby neighbours after applying the first stage of dynamics. All other extensions have their limitations when applied to

tracking many similar targets. It was considered that testing the *original* Condensation would provide an illustrative foundation of the degree of success offered by one of the most common tracking algorithms without any of the problem-specific bolt-on functionality offered by the extensions, and because of its ability to represent and process non-Gaussian probability densities it is suitable as a benchmark despite its limitations.

The MCMC algorithm with the interaction function (Khan et al. 2004) will be tested as this algorithm seems very suited to this work. Testing this algorithm should help quantify the failings of traditional Condensation, as well as providing state-of-the-art tracking results for groups of animals. Together these results should provide useful success and failure information for developing further algorithms for tracking multiple animals.

4.5. Video sequence capture

4.5.1. Pilot video: shot on location at a local farm

Prototype video of ducks was captured during January 2003. This footage was captured onto digital video from a camcorder on a static platform in the centre of a field of ducks on a farm site. The camera angle was low compared to the ideal angle – 'ideal' would be overhead as far as possible as this would provide the least occlusion opportunities. The practical lower angle was due to the limited height of the platform. However, general behaviour of groups of interacting ducks was observed, and preliminary image analysis techniques could be explored. This phase of the work also provided an estimate of the length of videos required. It was seen that the animals spent a large portion of their time resting, motionless. Most trackers will be able to maintain track of a stationary target over long periods of time as the dynamics of the target are not changing; it is in periods of action that tracking algorithms' motion processes are challenged the most. Therefore, using shorter, action-oriented sequences of ducks would typically provide the greatest challenge to the tracker. This is thought to be more meaningful than tracking using a longer, less challenging sequence with all the ducks motionless a large part of the time, which might suggest a tracking algorithm is more successful than it really is.

By both observing this video and by having first-hand experience with the animals for a substantial period of time, a number of different types of duck behaviour were found to reoccur often and thus be suitable for testing tracking algorithms. Some of these motions included:

1. Ducks gathered together in close proximity, with a small or no velocity. This happens in such occasions as feeding.
2. Ducks spread out with low or zero velocity. When resting, the ducks can be spread out throughout the enclosure and obviously have a very low or typically zero velocity. This situation happens a lot, but any tracker capable of handling more complex interactions should handle these occasions with ease, as there is little chance for tracking to fail.

3. When startled, the ducks can move in close proximity and with a high acceleration and final velocity. Such flocking can occur spontaneously, often from rest, and presents one of the hardest challenges to tracking.
4. Another common situation is for multiple groups of ducks to just 'amble' around the arena at medium to low velocity, moving generally as a group but also changing places within the group.
5. Another suitable test for a tracker is when two or more ducks move on different trajectories which cross, bringing the animals in close proximity for a short period of time.

A monitoring system would have to cope satisfactorily with tracking under all the above conditions, therefore these types of sequences will be the basis of the real-life test sequences used in the rest of this thesis.

The pilot video capture was initially designed to provide some testing video for preliminary experimentation with tracking methods, as the possibility of having ducks at Silsoe Research Institute was delayed until the summer months. However, because of the occlusions caused by objects in the arena and the low camera angle, this video really only served as an initial chance to observe ducks, providing the list of typical duck motions listed above. It is the aim of the final video capture to record such behaviours in a more controlled environment, with a higher camera than was available at the farm and with no occluding clutter between the camera and the animals.

4.5.2. Experimental setup at Silsoe Research Institute

In order to capture the required video for the experimentation, an outside enclosure was used on location at Silsoe Research Institute. This measured approximately 15m x 12m, and was surrounded by a fence. The ground cover was recently cut back to provide a short-grass type of covering with no potentially-occluding plants. In one corner a pond was placed, and in another a shed with bedding material. Also housed in the shed were the electrical points required to power any necessary equipment. Towards the centre of this enclosure,

a feeder and water trough were placed, as well as some boards by the fences to provide shade for the animals. A sprinkler was placed in the enclosure from time to time.

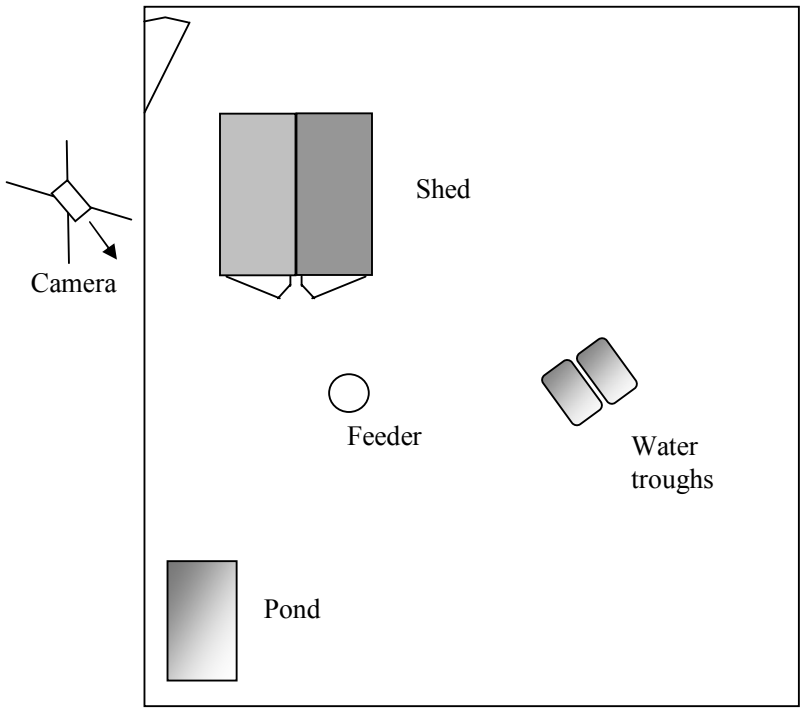


Figure 4.7 Representation of the experimental enclosure at Silsoe Research Institute.

As can be seen in Figure 4.7 and Figure 4.8, the camera was placed on a mast outside the enclosure. It was decided to place the camera outside the arena in order to give the widest possible view of the enclosure without specialist, heavily distorting lenses. Access was only possible to this side of the arena, and so the camera position illustrated was chosen. The camera was mounted on a telescopic mast approximately 6m high, and secured with guide ropes to prevent as much swaying motion as possible. Power was drawn from the shed, and the video output was sent to a neighbouring building approximately 10m away. S-Video cable was used to provide the greatest quality of output available with the given camera. Initial trials indicated that running this video cable into the neighbouring building resulted in a noticeable loss of signal integrity. Therefore, for the trials, a VCR was used as near to the mast as possible, resulting in much clearer images. The recordings were made to S-VHS tape, as with the available technology this

provided the highest quality images for the greatest lengths of time: 3 hour tapes could be used to capture as much activity as possible. A Pulnix PEC3010 colour camera was mounted in a protective cover and used for all the trials.

Figure 4.8 View from the animal enclosure showing the shed and camera mast

4.6. Image plane to the world plane coordinates

The detection of activities and recognition of events can take place in either the image plane or on a ground plane projection. Using the image plane removes any errors inherent in the coordinate transformation process when back projecting onto the ground plane. This is in line with previous work which has made the same assumptions (Johnson and Hogg 1996). For this work, where automated monitoring is the goal, it is advantageous to be able to set the system up in the difficult environment that is inherent in agriculture with a minimum number of pre-calibration steps. Although the enclosure used here has a certain known and visible geometry that could be approximated and used as a calibration guide itself (Liebowitz and Zisserman 1998), in other situations there may be no such features present. Lens properties may need to be altered for each individual situation depending on the field of view required, and so standard calibration values would be of no use. It is always possible to manually place calibration targets in the scene, but for this work it was considered unnecessary. Therefore, for as far as possible this work will use an uncalibrated image.

4.7. Determining the measurement model

For any kind of tracking algorithm, some sort of measurement model must be used to identify potential targets. These models tend to take the form of a function of an image-derived measure that is maximal when a target is present at the location being tested. Both contour and non-contour based functions can be used. A function which includes contour information can, like the snakes described in the previous chapter, impose some physical constraints on the geometry of the boundary, and include some sort of measure of the suitability of the boundary shape into the measure. Non-contour based measures make use of other features of the image to determine the presence of a target. Colour is one such feature that is commonly used (Nummiaro et al. 2003). Colour information can be used to derive a measurement for a colour-model based tracker. A maximal contrast is desired between the colour of the object to be tracked and the colour of the background over which the object passes. A more effective discrimination between the foreground object and the background can be made by selecting a suitable colour space in which the parameters of the target object and background are highly separated. There are several common colour spaces in existence, each of which has unique advantages and disadvantages. Hand selecting a space for a particular task can increase the efficiency and success rate of classification in that particular task, but does conversely decrease the generalisability of the work. It is the aim of this section to find a suitable type of observation measurement model, and then to determine a suitable colour space in which the model can most satisfactorily identify ducks against the background.

4.7.1. The observation model

An observational measurement model for this work is required to produce a maximal response when centred over a duck in an image. A typical image is presented below:

Figure 4.9 An example frame in the captured sequences. Notice how the contour changes depending on what state the duck is in: the marked ducks are likely to be preening, resting or feeding.

From this example, it can be seen that the contour formed by the outline of the duck is highly variable, and is both related to orientation and to the behaviour of the duck at that particular moment. Some of these ducks have ambiguous contours (Figure 4.9, arrowed) where some landmark features such as the head or tail are not visible. Typically, when eating or preening this contour can change rapidly and in a way unrelated to orientation; the head, for example, will appear and disappear as the duck carries out its task. Additionally, as the ducks are such small targets, the contour would not be able to identify many useful features to differentiate the ducks from each other. For these reasons, tracking the contour was thought to not yield enough useful or reliable information to be viable. Using colour as an alternative measurement model was thought viable due to the consistency in appearance of the ducks.

Previous work (Nummiaro et al. 2003) used pixel colours within an ellipse to successfully track various different classes of objects, from faces to cars to footballers. It was decided to implement a circular shape for the measurement model. Although an ellipse might more fully capture the body shape of some ducks, ellipses were not used in this work for simplicity and to maintain a common suitable model across all targets. Using a circle automatically crops the features (such as the head, neck and tail) that are subject to change depending on the orientation of the duck or its current actions, e.g. Figure 4.10:

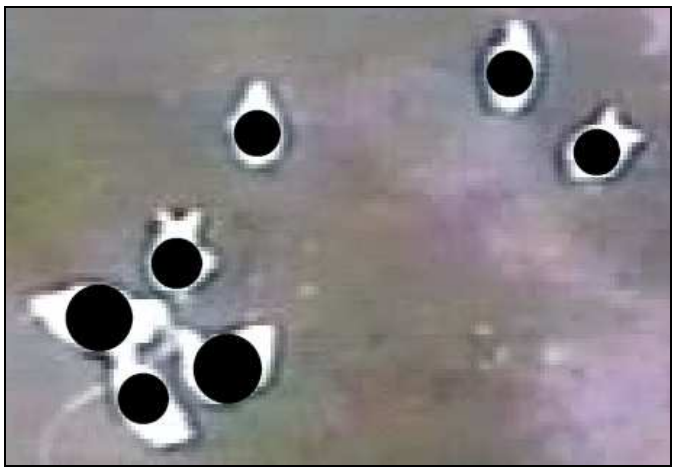


Figure 4.10 Circles provide a good fit to the body of a duck. Some duck bodies, e.g. the lower left duck, would be more suited to an ellipse model of shape. However, this would over-specialize the target model, and it would not fit other targets well, e.g. the rightmost duck.

An ellipse would be a more accurate way to model a duck shape viewed directly from above. However, from the camera angle used in this work, the silhouette of the ducks varies depending on the location of the duck and the duck's orientation relative to the camera. Therefore, it was considered that a circle would allow a more general posture- and viewpoint-independent measure to be made, and would also allow the dimensionality of the sample set to be reduced as the increased number of parameters required to describe a rotating ellipse would not be necessary. Increased dimensionality means increased sample numbers are required to represent the state space in algorithms such as Condensation. However, one advantage of using an ellipse would be to allow the orientation of the duck to be estimated: this could be used in future work with a fully overhead camera, where the duck outline is more predictable.

The pixels in the centre of the measuring circle were given more weight than the pixels towards the circumference as the further the pixels extend from the centre, the more likely they are to not represent the object being measured. Essentially, the pixels in the centre of the circle are more likely to stem from the target itself, as the radius may be over-estimated or some of the background may be visible on the periphery through an irregular duck contour. The pixels' weights in the measurement (how much they contribute to the colour measure) decrease in proportion to their squared distance from the circle centre (Nummiaro et al. 2003), as illustrated below:

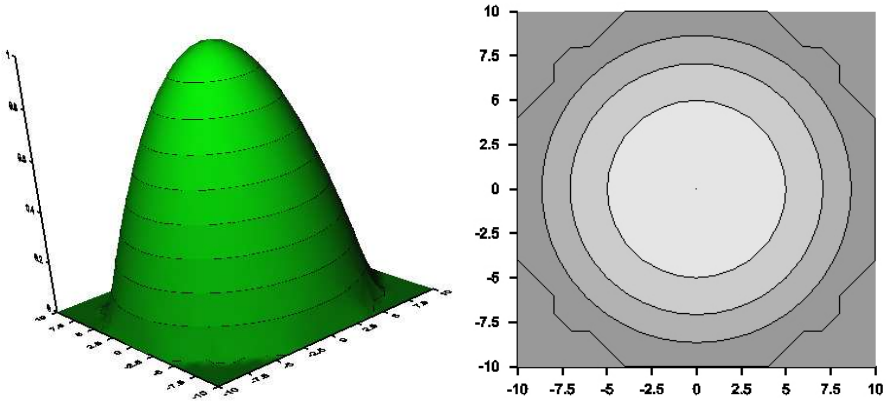


Figure 4.11 Graph to show how weight of pixels (Z-axis in the *left image*, intensity in the *right image*) varies across the target measurement circle (XY-axes)

The effectiveness of this method of measuring a weighted average colour is illustrated below using an example duck target from a test image.

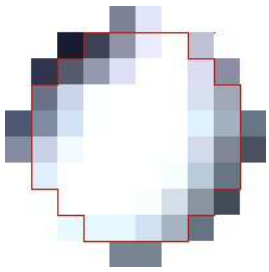


Figure 4.12 An example measurement circle of radius 5. The effective pixels, which have a non-zero weight, are bounded by the red line. This image is taken from a duck in a typical sequence.

Using a weighted circle for this measurement is preferable to point sampling a pixel as the best measurements occur when the circle is centred over the brightest pixels, whereas many individual pixels within the duck's boundary would give a good measurement if point-sampled. Using the circular model reduces the number of viable measurement positions generated by a target, and should position the tracker more reliably over the centre of the target.

Weighting towards the centre also prevents the measurement from being skewed by the occasional background pixel on the perimeter of the circle. For example, the circle in Figure 4.12 produces a weighted mean intensity of 233. A standard, un-weighted mean would give a measure of 199 – a value which has been dragged down by the dark background. If the measurement circle is made smaller to a radius of four, using the same data a weighted mean of 249 is produced, versus an un-weighted average of 243 that again has been lowered by darker background or shadowed edge pixels. There is less of a change with the smaller circle as less background pixels are erroneously captured. Weighting towards the centre allows for a more stable measurement across various postures, radii and different animals.

4.7.2. Determining Target Radius

As the image plane is not parallel to the ground plane, the radius of the circle used to take the colour measures has to vary as the size of the duck varies on the image plane. This is because targets further from the camera appear smaller than targets nearer to the camera. Initially, the radius was added into the state space of the samples, and was allowed to vary each timestep. This was found to be unsatisfactory. With no further constraints, the circles tended to reduce in size as there was no measurement benefit to filling the whole duck shape. A force was added to try to increase the size of the circles to prevent them from shrinking, similar to the inflating force used for some active contour models (Cohen 1991). However, this was found to be very difficult to balance, with the circle either expanding past the boundary of the target or continuing to shrink to a point. Excessive extension was a particular problem when two ducks were close to each

other. A measurement circle for a particular target would tend to expand to cover both targets, see Figure 4.13:

Figure 4.13 An unrestricted circle model can expand to cover the largest available white area, in this case two ducks instead of one

A different approach was adopted. As the radius varies in a largely deterministic manner across the image plane, a model could be used to set the size of the radius based on the image coordinates of the target. Although this method is dependant on the viewpoint, it is assumed that if the system were deployed for real, a similar camera setup would be used – therefore the technique described in this section, although specific to this type of camera rig, is general enough to cover expected real world equipment configurations. Assuming the same lens, mast height and camera angle were to be used, and the ground was flat as in the arena used here, the model for radius change can be used without modification. To calibrate for a setup where these parameters have changed, duck sizes across the image would have to be measured manually once more, perhaps using a custom calibration tool, and the model parameters re-calculated.

The short-radius (across the body) of 50 ducks at different locations in the camera image was measured. The image x-coordinate data did not on inspection correlate well with the radius size (see Figure 4.14). Image y-coordinate location followed what appeared to be a linear relationship with the radius size. This correlation was expected as the size of the target varies with distance from the camera, and therefore should be most apparent in the y-direction as distance increases at the highest rate per pixel in this direction, directly away from the camera. Therefore, a linear model was fitted to the image y-coordinate data to try and predict radius size from the y-coordinate (see Figure 4.15).

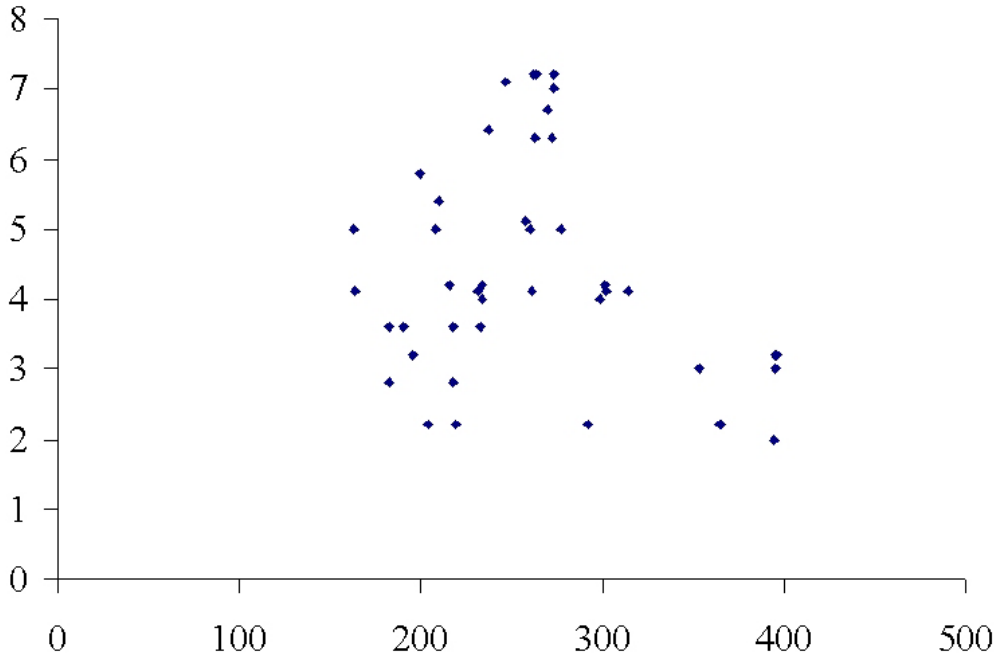


Figure 4.14 Image x-coordinate (x-axis, pixels) plotted against duck short-radius (y-axis, pixels). There is no apparent relationship between the two variables.

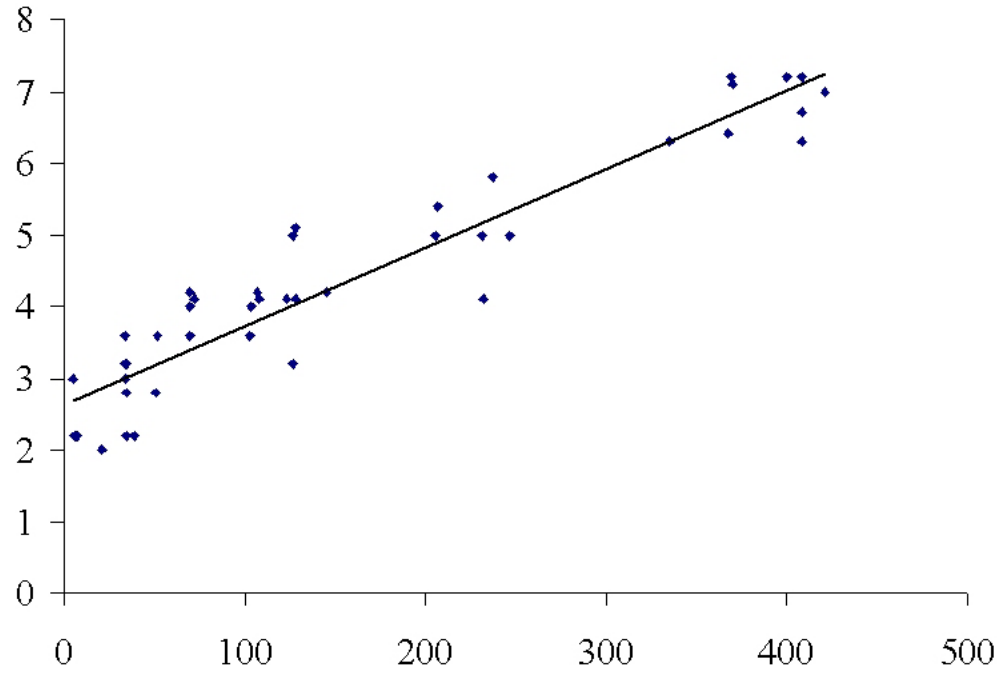


Figure 4.15 Image y-coordinate (x-axis, pixels) plotted against duck short-radius (y-axis, pixels). Equation of the line: $y=0.011x+2.7$, $r^2=0.89$.

The linear model of the radius derived from the y-coordinate accounts for a good amount of variation in the data ($r^2 = 0.89$). This model was then used in the measurement phase, to determine the radius of the measurement circle from the position of the duck in the image. Using a pre-determined target radius for each point in the image has the advantage that at any point it should only allow one target to fit underneath it: it will not expand to cover two targets as the state-space radius method did. This model is also advantageous to converting the image plane features to the ground plane, as no calibration features in the image are required other than the ducks themselves. In fact, using the targets themselves to calculate a ground plane conversion is a possible future extension of this work (Bose and Grimson 2003).

4.7.3. Colour space determination

When captured, the images are in the red-green-blue (RGB) colour space. Hue, Saturation and Value (HSV) space was examined as objects can be segmented using just hue and saturation – discarding value (intensity) makes the colour space more illumination invariant than RGB space. Following a similar argument, segmentation in chromaticity coordinates was considered. This is a normalised pure-colour space – again, intensity is effectively discarded and so the system is less susceptible to illumination change.

A comparison of the colour spaces was conducted to enable one to be selected that satisfies the following criteria:

1. It should be easy to differentiate the ducks from the background
2. The space should be fast to compute from the initial RGB space

The three spaces chosen to be compared are RGB (because it has no computation time and is widely used), HSV and Chromaticity (because they are more illumination invariant than RGB space). For each space, the level of distinction between the ducks and clutter is measured.

Chromaticity coordinates were assessed first. Tristimulus space (RGB in this case) is normalised to chromaticity coordinates using the following equations (Wyszecki and Stiles 1982):

$$r = \frac{R}{R+G+B} \quad g = \frac{G}{R+G+B} \quad b = \frac{B}{R+G+B} \tag{6}$$

It follows from (6) that $r + g + b = 1$, and hence that only two coordinates are needed to specify the entire space. At the expense of this, information on luminance has been lost. For example, take the RGB triples for an identically coloured object under different brightness levels, (10, 40, 15) and (20, 80, 30). When converted to chromaticity values, these would produce the same coordinates, (0.15, 0.62, 0.23). This can be used as a method of removing the issue of varying illumination levels: an object should have the same chromaticity coordinates no matter how brightly it is illuminated by a particular light.

Typical values of duck pixels in RGB space tend to (255, 255, 255) – the ducks are white and brightly illuminated and so tend to be recorded as a maximal signal on the camera sensor. Inspection of typical frames (see Table 4.1) reveals that the surrounding background (muddy grass) has a relatively wide range of values in RGB space but with one common attribute: the levels of each of the three RGB channels are very similar for a particular sample. This suggests that the chromatic value of the background does not drift far from the grey line in the RGB cube. The consequence of this is that when converted to chromaticity coordinates, the ducks and background appear very similar, and so it is not sensible to represent the images in this way.

In HSV space, similar problems are encountered. The hues of the background are not very saturated and so do not present much difference from the ducks. The lack of any high saturation levels (ducks and background are typically less than 10%) suggest that there is little useful colour information in the images. HSV space also requires more complex computation than chromaticity. However, HSV space can be made less sensitive to lighting conditions by placing less emphasis on the V value.

In RGB space, the objects represented are brightness dependent. As the level of light falling on the objects changes, so does the values of the red, green and blue channels. In all the test videos, the weather is such that the illumination conditions change slowly, as the sky is either clear or overcast. Under such conditions, an adaptive colour model could be used to keep the model up to date with slowly evolving illumination conditions (Nummiaro et al. 2003). This removes much of the need for illumination invariance. Additionally, the targets tend to largely saturate the sensor anyway and so the pixel values do not change with subtle weather effects.

Table 4.1 below presents summary colour information taken from typical frames in captured duck sequences. It can be seen that the ducks appear much brighter than their surroundings in RGB space because of the white appearance of their feathers.

	Mean RGB	RGB Range	Mean S,V (%)	HSV range
Ducks	253, 255, 255	33, 7, 0	1, 100	193, 13, 0
Background	119, 119, 116	118, 96, 122	8, 48	312, 13, 46

Table 4.1 Figures represent averages and ranges taken from 20 sample locations extracted from typical video frames. Note the very low saturation values for the ducks and the high brightness of the ducks compared to the background. No mean value for Hue is presented due to the non-linearity of the scale.

One physical effect worth mentioning here is when shadows are cast on the ducks. Occasionally clutter in the arena, such as the shed, produces shadows which the ducks sometimes walk through.

Figure 4.16 The effect of a shadow on a duck. In the sun, the duck's RGB values are (255, 255, 255). When it walks into the shadow, a typical pixel is (168,220,255). Note how the red channel has the largest decrease in value.

Such an effect tends to reduce the value of the red channel, though the blue and green channels tend to remain high. This process can be handled by increasing the standard deviation of the expected measures in the red channel, to account for the increased variability.

It has been shown that neither using HSV space nor using chromaticity coordinates provides an especially useful environment for segmenting and locating ducks. Both of these spaces also require calculation time to convert from the original RGB image. RGB provides a good separation between the ducks and the background, based largely on intensity because the ducks appear very bright compared to the background. Although the HSV and chromaticity spaces provide a more illumination independent framework, RGB values of the ducks do not change enough to warrant the processing overhead of such a system. Using RGB it should always be possible to satisfactorily discriminate the ducks from the background.

4.7.4. Building the colour model

A three-channel RGB colour model was therefore used as a way of determining how similar a measurement was to an ideal. The colour model was built by measuring the colour of ducks in a number of sample frames. The RGB values of 100 duck images from seven different sequences was measured, this time using the weighted circle measurement model. These measurement circles were manually centred on the estimated centre of the ducks, and the radius of the circles

was automatically determined according to the location on the image, as used in the program itself (see section 4.7.2, above). The mean of these 100 RGB duck colour values was used as the ideal values of a duck for each channel of the duck measurement model. The mean duck colour measurements were found to be:

Red	247.5
Green	250.8
Blue	252.0

The measured values were then normalised so that they fall between zero and one, zero representing no value on a channel and one representing a saturated value of 255. This was done for ease of possible future extensions, where measurements may not fall between 0 and 255. The standard deviations of these values were used to estimate the parameters for the Gaussian observation model. These standard deviations were found to be 0.039, 0.034 and 0.027 respectively for the three RGB channels. However, using these values, initial experiments showed tracking performance to be poor. With some sequences, these values did not account for the level of variation in the data, and measurements were too low with the ducks in some lighting conditions, for example the differences in weather conditions produce some variability. This caused the tracker to fail more often, typically by drifting off the target. Further investigation showed that the standard deviation *within* certain sequences, even those included in the 100 measurement sample set, was actually greater than the values calculated for the complete set of 100 measurements. It was therefore necessary to increase the standard deviation to account for the extra variability inherent in individual sequences. The standard deviation ranges for individual sequences was between 0.0098 and 0.089 for the red channel, 0.0025 and 0.077 for the green channel and 0.0018 and 0.065 for blue. Therefore the standard deviations for the sequences were increased more in line with the maximums of these ranges, plus some extra amount determined by trial and error to account for further variability in the actual data. Values of 0.12, 0.10 and 0.08 for the red, green and blue channels respectively were found to work well in practice.

It is noted that this method of determining the colour model is rather ad-hoc; however, this is the procedure that was used and so is described here for completeness. Given the opportunity to perform the model development again, a number of things would be taken into consideration. First, whether the data actually is Gaussian; given that the ducks can saturate the sensor on occasions, this assumption may not hold throughout all the sequences and so should be investigated. Second, more atypical examples would be added to the training set, as it is just these kinds of situations which cause the tracking to be lost when the standard deviations of the colour model are set too low. Considering these additional factors should provide a more robust method of determining the standard deviations, which should in turn produce more reliable tracking without having to adjust these parameters by trail and error.

One additional problem is what happens to the colour measurement when the duck is in shadow. In such a situation, the measurement from the red channel differs greatly from a similar type of measurement in direct sunlight. It is the red channel that is most affected because hues from the cooler end of the spectrum (i.e. blues and greens) are diffused from the blue sky and are less affected by the shadow, whereas the direct red frequencies from the sun are blocked out by the shadow-inducing obstruction. Scaling the red channel's standard deviation by a larger amount allows for this extra variability in the red channel incurred by shadows (Figure 4.16 on page 125). The exact amount is hard to quantify as it depends on the number of shadows in the scene, which varies depending on the time of day and the strength of the sunlight. However, scaling this channel by a further factor of three has been found to produce very good tracking results. For future work, it may be possible to refine and justify this parameter by looking at existing literature on shadow detection and elimination, but currently this estimate seems to work satisfactorily. It is also noted that this large rescaling may not be necessary in future work if the colour model parameters were to be determined more theoretically, as described in the previous paragraph. For simplicity and processing speed, rather than evaluate a 3D measurement density in RGB space the final density is simply constructed as the product of the 3 independent densities for each of the three colour channels (Blake and Isard 1998).

4.8. Tracking ducks using the Condensation algorithm

4.8.1. Introduction and problem

It is the purpose of this experiment to determine how well standard Condensation can track multiple ducks in an outside arena, in order to provide a benchmark for subsequent experiments. Condensation (Isard and Blake 1996; Isard and Blake 1998b) is an established way of tracking objects when high levels of clutter are present. For all the reasons given in the introduction to this chapter, such as its ability to handle tracking in clutter, Condensation was considered as an appropriate candidate algorithm to evaluate. Condensation tracking has been used in past work to track targets using colour models (Nummiaro et al. 2003), and this work will implement a colour observation model as specified in section 4.7.

This experiment will use multiple independent Condensation trackers, one for each target. Maintaining each target's parameters inside one joint state space leads to very high sample numbers being required, which in turn leads to very slow processing times. This is because of the dimensionality problem of importance sampling. The joint space can be approximated by using multiple independent trackers (Khan et al. 2003), and although it is recognised that this may not perform as well as an optimal joint state space tracker, it will serve to quantify and illustrate the problems of multiple independent trackers. The joint state space is introduced along with more efficient Markov chain Monte Carlo sampling in section 4.9.

4.8.2. Implementing the algorithm

The algorithm is presented below, and an explanation follows.

CONDENSATION ALGORITHM

1. **Randomly select** N samples $\{X_t^{(r)}\}_{r=1}^{N}$ from the sample set from last timestep, $\{X_{t-1}^{(r)}, \pi_{t-1}^{(r)}\}_{r=1}^{N}$, with probability proportional to their weight, $\pi_{t-1}^{(r)}$

2. **Propagate** sample set forward by sampling from: $p(X_t \mid X_{t-1})$

3. **Measure**. Weight each particle based on a measurement using the colour model: $\pi_t^{(r)} = p(Z_t \mid X_t^{(r)})$

Algorithm 4.1 Condensation algorithm (Isard and Blake 1998b)

The following sample statespace was used:

$$X = (x, y, u, v), \pi$$

where x and y are the location coordinates of the target, u and v are the velocities of the target, and π is the weight of the sample.

Each time-step is a self-contained iteration of factored sampling, hence the output at each timestep is a weighted sample set. The prior at each time step is the measurement-weighted sampleset from the previous timestep. This is a potentially multi-modal distribution. The distribution represented by this set is processed forward in time to provide a prediction using the motion model. This model has two elements: a drift based on the previous motion of the particle and a random diffusion element. This new, unweighted sample set is then weighted by taking measurements for each of the samples using the observation model, and weighting them accordingly. This process is repeated, using a fixed number of samples at each timestep.

PARAMETER CHOICE

A number of parameters control the functionality of the Condensation algorithm: the process noise, measurement model parameters and the initial values in the statespace. Measurement model details have already been presented; the following section details the choice of process noise parameters and the initial conditions.

Process noise is applied to the state parameters of the tracker to allow them to change over time. One of these parameters is position, which in turn allows the movement of the target to be modelled. Translation of the target in the (x,y)-plane differs from that predicted by the velocity due to three main factors: the 'wobble' due to the duck's gait, the wind blowing the camera and the acceleration of the duck. Acceleration is covered separately as noise on the (u,v) parameters (i.e. change in velocity); see below.

The (x,y) noise must account, then, mainly for the wind effects on the capturing equipment and random effects on the duck's motion gait (as velocity is handled by separate parameters, explained below). As this positional error is the result of multiple independent error effects, its distribution can be assumed Gaussian. This effect can be observed to produce approximately 1 pixel of error every second or so. This value is hard to measure reliably and so is based on manual observation and familiarity with the sequences. Per frame, this error works out to be 0.04 pixels. To keep 95% of noise within this range, the value for the (x,y)-noise standard deviation should be $0.04/1.96 \approx 0.02$ (as 1.96 standard deviations from the mean of a Gaussian curve contain 95% of data). This value is considered reasonable, as then nearly all of the noise is constrained to within observed values, with 5% of the noise falling outside this range, allowing for occasional extreme values.

The noise on the velocity (u,v) parameters is effectively the rate of change of velocity, or the acceleration. Measuring empirically the typical acceleration values of a duck is a challenging prospect. After much trial and error varying this parameter, a standard deviation of 0.15 pixels per frame was used. This means

95% of velocity noise falls between 0 and 7.5 pixels / second^2. From manual observation and testing, this is found to be adequate for most situations. It is noted that strictly this parameter should vary in relation to a duck's position on the image plane, as a duck further away from the camera on the ground plane will appear to have a lower pixel acceleration than a closer companion. However, as long as the maximum acceleration expected is covered by this noise, then this simplification is considered acceptable.

With Condensation, the number of samples used to represent the system can be varied. Higher numbers of samples lead typically to increased tracking performance but also to increased processing time. To enable a comparison between algorithms and sequences, the number of samples will be kept constant per target in the image; 300 samples per target was found to produce a good tracking result in the challenging real-world environment.

INITIAL STATE VALUES

For each sequence, the initial positions of the targets are measured by hand. The coordinates of the centre of the ducks are measured from an image manipulation program. Initial samples are automatically spread around this location, with an (x,y)-noise standard deviation of 5 pixels. Therefore, 95% of samples should be placed within about 10 pixels of the manually entered positions. This is to account for any manual inaccuracy in measuring the starting locations. On a real world system this initialisation could take the form of a user clicking a live video stream of the ducks to provide the locations of the targets. As the (x,y) locations of the targets are entered into the program, the tracking could begin on the next frame of the video feed.

The initial velocities are set to zero, with a distribution equal to the standard deviation of the process noise on the velocities, 0.15 pixels per frame. In some situations where targets are already in motion, it is advantageous to set the initial velocity to represent this, though unless explicitly mentioned, it should be assumed the initial velocity is zero.

The colour measure parameters remain the same on each sequence.

4.8.3. Measuring success

In different situations, various different methods of measuring the success of a tracker may be applicable. A summary of various methods will be presented, followed by which methods were chosen for this work.

One of the most common ways to measure how well a tracker performs is to measure the error between the tracking output and some known truth, called a ground truth. This error is often presented as a root mean square (RMS) error for the sequence, a larger error indicating that the tracking output was further from the ground truth positions, and so can be considered less accurate. Measuring RMS errors would be a logical choice of success criteria where accuracy over the whole tracking sequence is important, perhaps for industrial applications where feature locations are required to be consistently very accurate. RMS errors are, however, a sensible measure only where the tracker never loses the target completely. If it *does* lose the target, the error becomes harder to interpret. The error will be much greater if it is lost near the beginning of the sequence than if it is lost towards the end. Also, RMS errors will penalise a tracker that loses a target for a small period of time, even though it does eventually regain tracking of the target. These kinds of errors may not be of interest if our main consideration is whether the *identities* of the targets are maintained throughout the sequence, rather than the actual *accuracy* of placement. Additionally, when the tracker has lost the target, the RMS error will be different depending *how* far off the tracker is: in reality does it matter if it is misplaced on clutter close to the target or on clutter on the other side of the image? In some situations this may matter, but when any loss of target identity has serious consequences, such as for monitoring purposes, how far wrong the tracking estimate is is less important to know than simply the fact it *has* misplaced the target.

Because of these effects, it is important to understand what is happening in the tracking results as well as just quoting an RMS error. Perhaps the best way to determine what is happening in the sequence is manual observation. Simply

watching what is happening to a tracker throughout a sequence can reveal details that other methods cannot. The main reason for qualitative observation of a sequence of results is to see what causes a tracker to fail. RMS errors, for example, do not offer an explanation as to which situations are most likely to cause tracking to be lost. The method of manual observation is applicable, therefore, to situations where the cause of error is important, perhaps to look for ways of improving a tracking system, by changing the algorithm or altering the domain in which the tracker works.

A third measurement of tracking success is to record the ability of the tracking system to maintain a target's identity throughout a sequence. This measure might simply be to see if a tracker finishes the test sequence on the correct target. This method is not concerned with the accuracy of the tracking placement over the target, or with the ability of the tracker to identify the target on every frame. Instead, what is important is that the tracker can maintain tracking of a target over a substantial period of time. Such measures become useful in domains where temporally lengthy sequences are to be analysed, where there are plenty of opportunities for the tracker to lose its target completely such as tracking an individual walking through a crowd. Disadvantages of this method include the inability to say whether tracking was temporarily lost at some point in the sequence, hence why this method must be combined with manual observation of the tracking results.

The final method considered here is related to RMS: measuring the residuals between the tracking result and the ground truth at each frame. RMS errors provide one value that summarises the tracking success over a sequence, whereas examining the residuals provides a measure of success on a frame-by-frame basis. Residuals can be used to determine where in a sequence a tracker went wrong, and also to see how consistent the tracking accuracy was. It can also be used to see if a tracker temporally loses and regains a target.

In this thesis, a combination of the above methods is used. A groundtruth for the sequence in question is created for the sequences under test. A combination of

residuals and qualitative manual observation is then used. Residual graphs are plotted, displaying the difference between the groundtruth and the tracking results each frame. Interesting events on this plot, such as increases in residual value (temporary or permanent) are then related to the action taking place in the relevant frames of the video. This allows an understanding of why the tracking went wrong. In monitoring groups of animals for this work, fine positional accuracy is less important than maintaining a target's identity throughout a sequence. The ability to trace a duck's approximate movements for a long period of time is considered more useful than very accurately locating its position for a short period. Therefore RMS errors are only used with this caveat, and only on tracking results where the target is successfully tracked through the sequence of interest, as measuring RMS errors where the tracker losses its target is meaningless for this work. The ability of a tracker to maintain an identity through to the last frame in a sequence is the most important criterion for success in this thesis.

CREATING THE GROUNDTRUTHS

Groundtruths for the particular test sequences were created manually using custom software. The user is prompted to click on particular targets in each frame, and this position is recorded to a log file. Due to time limitations and the time-consuming nature of this technique, this process was only completed once by one individual, rather than the ideal situation of generating multiple sequences by multiple people to prevent subjectivity effects. With any manually generated sequence, there are human errors which must be taken into account. Macro-scale errors were picked up by parsing the data for large changes in position from frame to frame, and flagging such events to the user for approval.

The operator must click the centre of the duck on the frame. Obviously to a degree this is a subjective operation and might vary from user to user and time to time. The groundtruth for a sequence of four targets over 50 frames was repeated three times with the same human operator to produce an estimate of the quality of the groundtruths. Variation in the placement accuracy of the groundtruths produced a standard deviation of 0.95 pixels. Therefore, most locations (68%) are

within about one pixel of their true value, and nearly every location (99% of them) within 3 pixels.

4.8.4. Results of tracking sequences

The Condensation algorithm was tested on 4 common situations that occur with groups of social targets. These situations can be seen by observing a video sequence of the ducks, and were initially observed when recording the pilot video (see section 4.5.1). They do not represent an exhaustive list of events, merely a representative example set that is challenging to a tracking algorithm:

1. Multiple targets with no obvious goal ("ambling")
2. High velocity, close proximity
3. Targets moving in different directions with crossing paths
4. Low velocity, close proximity

With the ducks, these general scenarios can be represented by the following specific instances of actions:

1. Ducks moving around slowly in a small group (sequence 1)
2. Ducks exhibiting flocking behaviour (sequence 2)
3. Ducks heading in different directions with crossing paths (sequence 3)
4. Duck eating from a common food source (sequence 4)

Sequences were found that exemplify these scenarios. Opening frames from these sequences are presented in Figure 4.17, overleaf.

Figure 4.17 Opening frames from the four sequences, relating to scenarios 1-4 (from top left.) Ducks uninvolved (i.e. a suitable distance from and uninvolved with) the targets were not tracked during this experiment.

Results of standard condensation tracking these four sequences shall now be presented. Unless otherwise stated, the Condensation tracker uses 300 samples per target.

Sample locations are presented as dots, and on this Condensation experiment the samples are shaded from black (low weight) to the tracker colour (high weight). The best estimate location calculated for these samples is represented as a circle, calculated as a weighted mean.

4.8.5. Sequence 1 results using Condensation

Figure 4.18 Groundtruth paths for sequence 1

Figure 4.19 Result paths for sequence 1

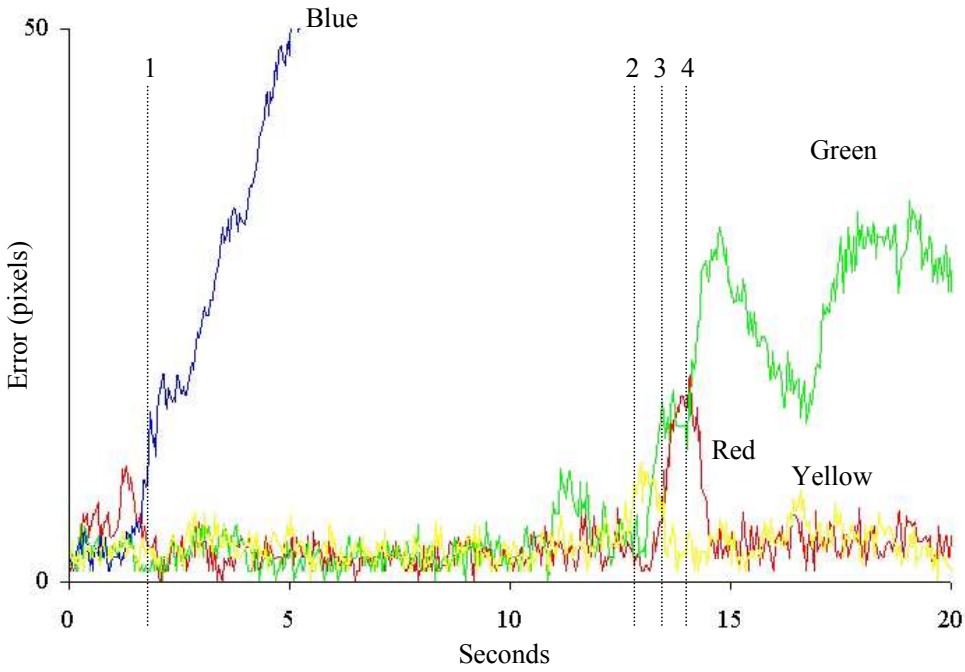


Figure 4.20 Residuals of groundtruth and actual data for sequence 1, with interesting frames marked with vertical bars. These bars represent (left to right): (1) blue migrates to green's target, (2) yellow is temporarily lost off the back end of the target (3) red and green split between 2 targets (4) red confused and temporarily misplaced.

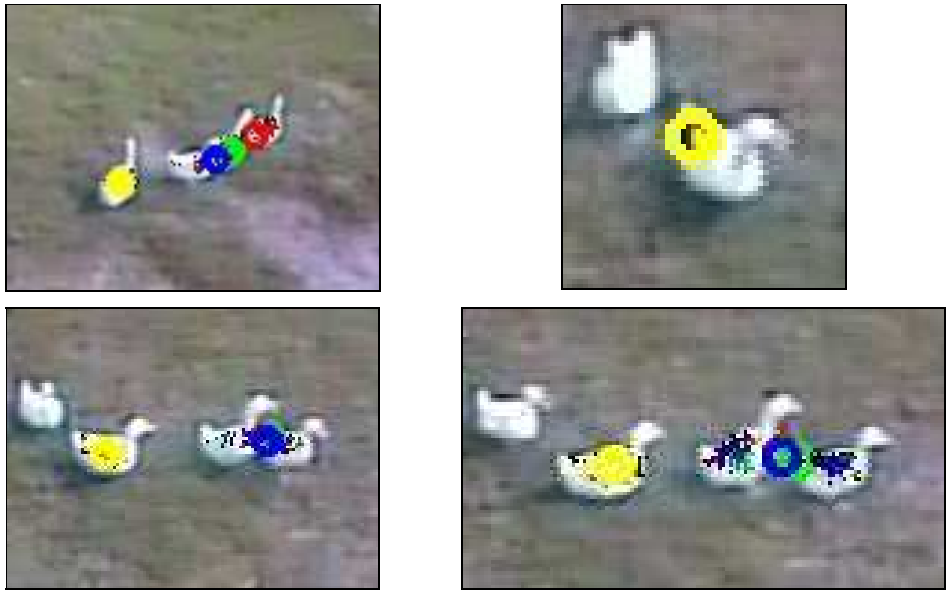


Figure 4.21 Output frames relating to the time markers in Figure 4.20. From top left, clockwise: (1) blue migrates to green's target, (2) yellow is temporarily lost off the trailing end of the target (3) red and green split between 2 targets (4) red confused and temporarily misplaced. Note how in (3) and (4) particularly, the samples are spread across multiple targets.

The groundtruth for the sequence can be seen in Figure 4.18. This is a complicated sequence to track because the ducks are moving close to each other, and occasionally their paths interleave. It can be seen from the resultant paths in Figure 4.19 that 3 of the trackers – red, green and blue – all finish up following the same duck (the red target), which must provide the highest measurement response. Therefore, the green and blue trackers end up misplaced. The yellow target is tracked successfully, despite almost losing track at one point, possibly due to another target being close, or because of the motion of the animal (see (2) in Figure 4.20 and Figure 4.21). The blue tracker loses its correct target early on, after only about two seconds when it migrates onto the green tracker's target (see (1) in Figure 4.20 and Figure 4.21). Despite some confusion at one point in the sequence (see (4) in Figure 4.20 and Figure 4.21) the red target's identity is successfully maintained after 20 seconds. The green tracker ends up in a similar situation to the red tracker (see (3) and (4) in Figure 4.20 and Figure 4.21), caught between two targets, but is unable to recover. Given the baseline noise, the successfully tracked targets (yellow and red) are tracked quite accurately throughout the sequence (Figure 4.20). This is apparent in the RMS error for the red tracker (4.6 pixels) and the yellow tracker (3.8 pixels).

4.8.6. Sequence 2 results using Condensation

Figure 4.22 Groundtruth paths for sequence 2

Figure 4.23 Result paths for sequence 2

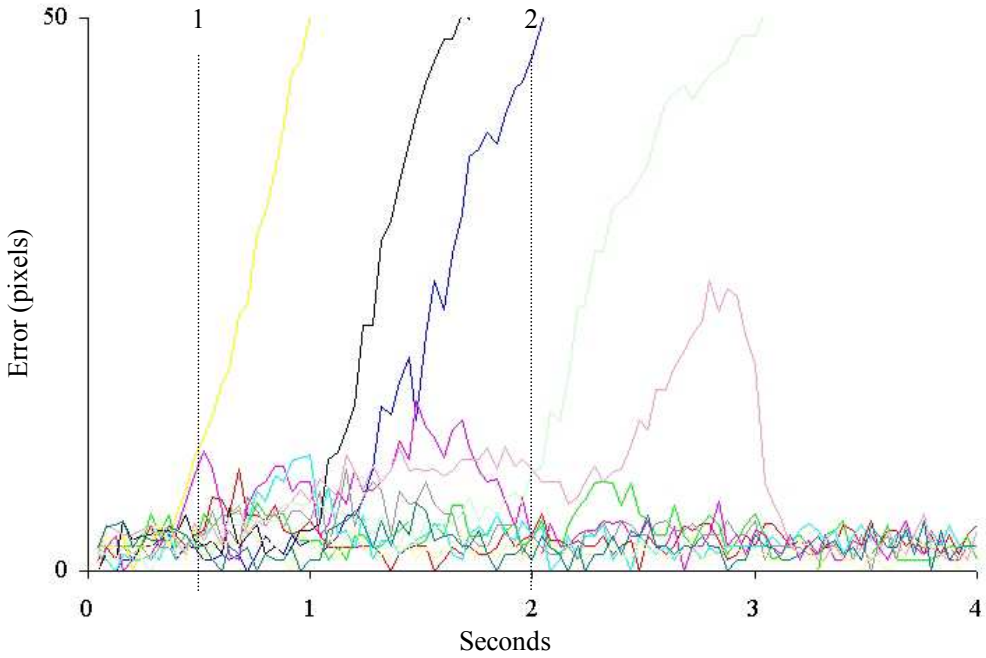


Figure 4.24 Residuals of groundtruth and actual data for sequence 2, with interesting frames marked with vertical bars. These bars represent (left to right): (1) Yellow attains incorrect target duck, and dark pink is left behind after rapid acceleration. (2) Yellow, black and blue trackers are now a long way off their intended targets.

Figure 4.25 Output frames relating to the time markers in Figure 4.24. From left to right: (1) The yellow tracker migrates onto the wrong target. Note that the dark pink tracker is left behind at the edge of the target due to the rapid acceleration of the target. (2) Yellow black and blue targets have been lost. Note that the dark pink tracker has regained the correct target (top-right of the frame)

The groundtruth for this sequence can be seen in Figure 4.22. The ducks make a fast flocking motion away from the water troughs, as if startled by something. This fast motion from near stationary initial positions makes this a hard sequence to track. Sometimes the tracker can cope with these rapid accelerations.

Following the pink tracker for example, at about 0.5 seconds the tracker is almost off the target due to the acceleration: this can be seen in Figure 4.25 (1). Because of the lack of clutter near the tracker, tracking is able to be resumed, as can be seen from the temporary peak in the pink residual line in Figure 4.24 (1).

Using residuals as a guide to tell how accurate a tracker is does have some limitations. For example, if two targets are close together and the trackers get the two targets confused, this is not always obvious when examining the residuals because of the proximity of the two targets. An example of this can be seen in Figure 4.24 (2) where the light pink line is off target, but because it is confused with a nearby target the residual line is not very high.

4.8.7. Sequence 3 results using Condensation

Figure 4.26 Groundtruth for sequence 3

Figure 4.27 Result paths for sequence 3

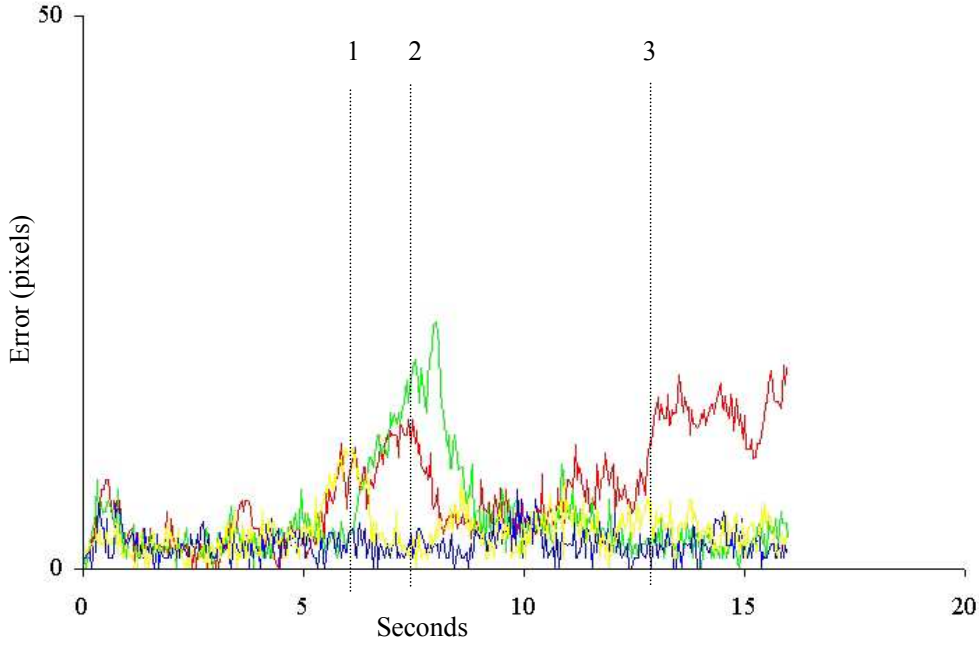


Figure 4.28 Residuals of groundtruth and actual data for sequence 3, with interesting frames marked with vertical bars. These bars represent (left to right) : (1) Yellow tracker is temporarily confused with the green and red targets, but because of the velocity differential tracking is easily resumed when the yellow target passes by. (2) The red and green trackers have switched targets (red tracker on green target and vice versa) and continue to remain ambiguous for a number of seconds until... (3) ...red samples gradually migrate irrecoverably to the higher quality green target.

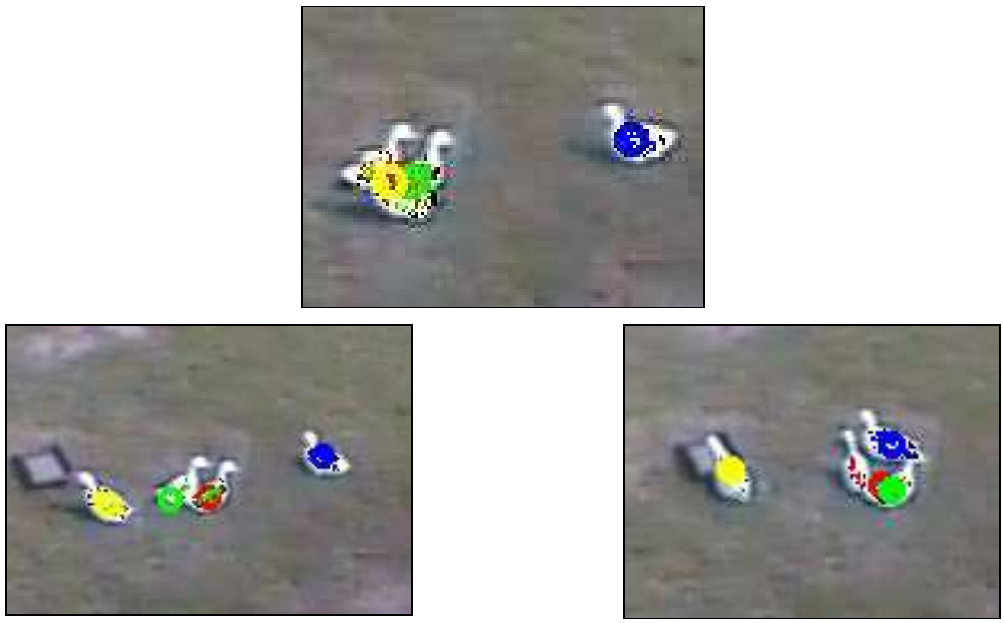


Figure 4.29 Output frames relating to the time markers in Figure 4.28. Top (1): The yellow target passes by two other ducks, and the tracker is temporarily attracted to them. However, as the yellow target is moving much faster than the two other ducks and in a different direction, correct tracking is resumed after a short while. Left (2): red and green trackers have swapped targets due to the proximity and similarity of the targets. Note that some green samples remain centred on the red target. This situation continues for about 5 seconds, during which the green tracker regains tracking but the red tracker migrates onto the green target (right, (3)).

Figure 4.28 shows that the Condensation algorithm during this sequence produces target estimations that are generally quite accurate. The red and green trackers do swap targets and sometimes double up on a target (Figure 4.29). This is because the trackers are not aware of the action of other trackers, hence they can share measurements. In fact, the tracking of the red and green targets remains ambigous between the (2) and (3) markers in Figure 4.28. However, because the ducks are sometimes in very close proximity, these errors are not always obvious from the residual graph. This is one problem with using this technique to detect errors, and is why this method should be used alongside qualitative observation.

The blue track is correctly maintained throughout the sequence (RMS of 2.5 pixels), although this duck has the least interaction with the others, offering fewer occasions for the tracker to be attracted to other targets. Overall though, the tracking can be considered a success for 3 of the targets; at the end of the sequence, 3 out of the 4 ducks are successfully located and identified.

4.8.8. Sequence 4 results using Condensation

This sequence consists of 400 frames in which a group of 11 ducks are around a water source. There is very little motion in the image, and this sequence test the tracker's ability to maintain the correct target when multiple similar targets are very close by. The best analysis for this sequence is qualitative observation, as due to there being almost zero motion the test is for the tracker to maintain target identity rather than path accuracy.

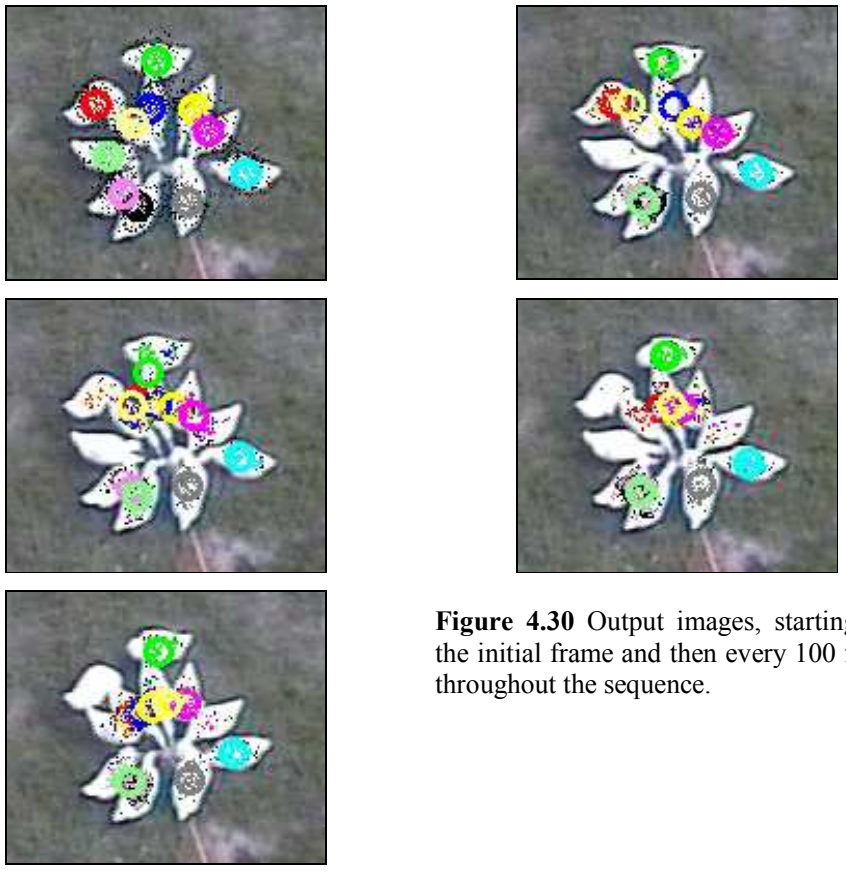


Figure 4.30 Output images, starting with the initial frame and then every 100 frames throughout the sequence.

It can be seen from Figure 4.30 that having multiple independent trackers unaware of each other's presence quickly leads to the trackers coalescing on the targets producing the best measurement response. The light green tracker (bottom left in the last three images) is spread across two targets. This is because the samples are divided between two targets and the estimation of target location is therefore in the middle of the sample cloud.

4.9. Tracking ducks using a joint state and interaction model

4.9.1. Introduction and problem

It has been shown that the tracking of multiple similar targets is a difficult task. The previous section's work using Condensation suggested that a model of interactions might be required to keep the trackers on target when the targets are in close proximity; having independent trackers with no knowledge about each other leads to the trackers coalescing on the best target measurement. The most promising existing algorithm for these scenarios in existing literature uses a Markov chain Monte Carlo method to represent the joint state of all targets, and uses an interaction function to model close proximity interactions (Khan et al. 2004). This algorithm will first be presented and then tested with the same sequences used to test Condensation, above.

MARKOV CHAIN MONTE CARLO ALGORITHM

1. **State of targets** at t-1 represented by a set of samples $\{X_{t-1}^{(r)}\}_{r=1}^{N}$. Each sample contains information on the complete joint state
2. **Initialize** the MCMC sampler at time t by drawing X_t from the standard (interaction-free) predictive density $\sum_r \prod_i P(X_{it} \mid X_{i(t-1)}^{(r)})$
3. **Metropolis-Hastings iterations.** Obtain M samples from the posterior. Discard the first B samples for sampler burn in. In detail:
   a. **Proposal:**
      i. Randomly select a joint sample $X_{t-1}^{(r)}$ from the unweighted samples from t-1.
      ii. Randomly select target i from the n targets in the joint state. This is the target to move this iteration.
      iii. Apply the motion model to this target to obtain $X_{it}^{'}$.
   b. **Calculate acceptance ratio:**
   $$a_s = \min\left(1, \frac{P(Z_{it} \mid X_{it}^{'}) \prod_{j \in E_i} \psi(X_{it}^{'}, X_{jt})}{P(Z_{it} \mid X_{it}) \prod_{j \in E_i} \psi(X_{it}, X_{jt})}\right)$$
   c. If a_s = 1 then accept the proposed sample $X_{it}^{'}$, i.e. set the ith target in X_t to $X_{it}^{'}$. Else if a_s<1 we accept the proposed sample with probability a_s. If rejected, leave ith target in X_t unchanged.
   d. Add a copy of X_t to the new sample set.
4. $\{X_t^{(s)}\}_{s=1}^{M}$ represents the estimated joint posterior.

Algorithm 4.2 Steps of the Markov chain Monte Carlo tracking algorithm presented in previous work (Khan et al. 2004).

X_t is effectively the current best-estimate sample. Therefore, as the number of iterations increases the quality of sample added to the new set should generally improve – hence the burn in phase which rejects typically the first 25% of samples (the poorest quality ones) before adding samples to the new set. The final set differs from Condensation in that it is unweighted – the distribution is represented by the frequency of samples rather than weights.

4.9.2. Coping with interactions

When tracking multiple *non-interacting* targets, running a single-target Bayesian filter for each target allows us to approximate a complete Bayes filter in a way that is computationally tractable. However this approach breaks down when targets start interacting. Typically, the trackers will all jump onto the target with the highest score. This has been observed with insect tracking (Khan et al. 2004), and with duck tracking in the previous experimental section with multiple independent Condensation trackers.

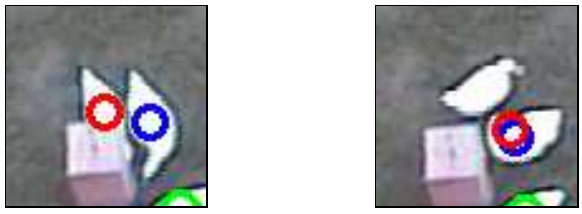


Figure 4.31 Left: two ducks come to feed. Right: approximately one second later, one of the targets has been 'hijacked' by one of the trackers. This problem occurs with multiple independent trackers where one target produces a higher quality measurement than the other.

Khan et al. overcome this problem by incorporating a Markov Random Field interaction factor into the motion model for the particle filter. This is added into a *joint state* filter, considering the states of all *n* targets . In order to represent the joint state efficiently, a different sampling approach is used: Markov chain Monte Carlo sampling (Khan et al. 2004).

The practical effect of the interaction model is an additional factor in the observation model for the particles; as well as being evaluated by a measurement from the image, the particles are also affected by the value of the interaction term, as in equation (7).

$$\overbrace{P(Z_{it} \mid X_{it})}^{\text{Image measurement}} \overbrace{\prod_{j \in E_i} \psi(X_{it}, X_{jt})}^{\text{Interaction metric}} \tag{7}$$

where E_i is the set of Markov random field graph edges connected to target i (i.e. targets with which an interaction can occur), and ψ is the interaction function. This interaction function takes the form of a Markov random field-based motion model, which produces a low probability score if targets are within a certain distance of each other. This prevents the tracker from allowing targets to occupy the same location, preventing the 'target hijacking' effects that multiple independent Condensation trackers produced.

As well as this interaction and the representation of a joint state, this algorithm's main difference from Condensation is the use of Markov chain Monte Carlo (MCMC) sampling. MCMC replaces the less-efficient importance sampling step as used in Condensation. This more efficient step generates a sequence of states which represent the target distribution. A Markov chain is defined such that the stationary distribution of the chain is exactly the target distribution. This sampling can be done using the Metropolis-Hastings algorithm, where new states are chosen based on a *likelihood ratio* between a proposed state and the previous state. The set of states therefore evolves in a manner similar to using a genetic algorithm, and proposed states are evaluated against a likelihood more than once per frame (unlike Condensation). It is this efficient sampling step that allows a joint state to be represented without the exponential overheads introduced with algorithms such as Condensation, where to represent a joint state so many samples must be used that the tracking is painfully slow. For ease of reference, this Markov chain Monte Carlo algorithm with the Markov random field interaction function will be referred to herein as the MCMC MRF algorithm

4.10. Results of tracking sequences

The algorithm was tested on the same sequences as used to test Condensation in Section 4.8. Unless otherwise stated, the same parameters will be used in this section, with the exception of the number of particles. For the number of samples, the number to represent the joint state will be kept equal to the total number for all the targets that Condensation used for a particular sequence. In other words the total number of samples in the system will be kept constant for each number of targets. This should keep the time to run approximately equal for the two algorithms. As with Condensation, coloured dots represent tracking estimates of target locations. Shading is not possible on the Markov chain Monte Carlo samples as they have no associated weight. The circles represent the mean location of the particles' estimates.

4.10.1. Sequence 1 results using MCMC MRF

Figure 4.32 Groundtruth paths for sequence 1

Figure 4.33 Results paths for sequence 1

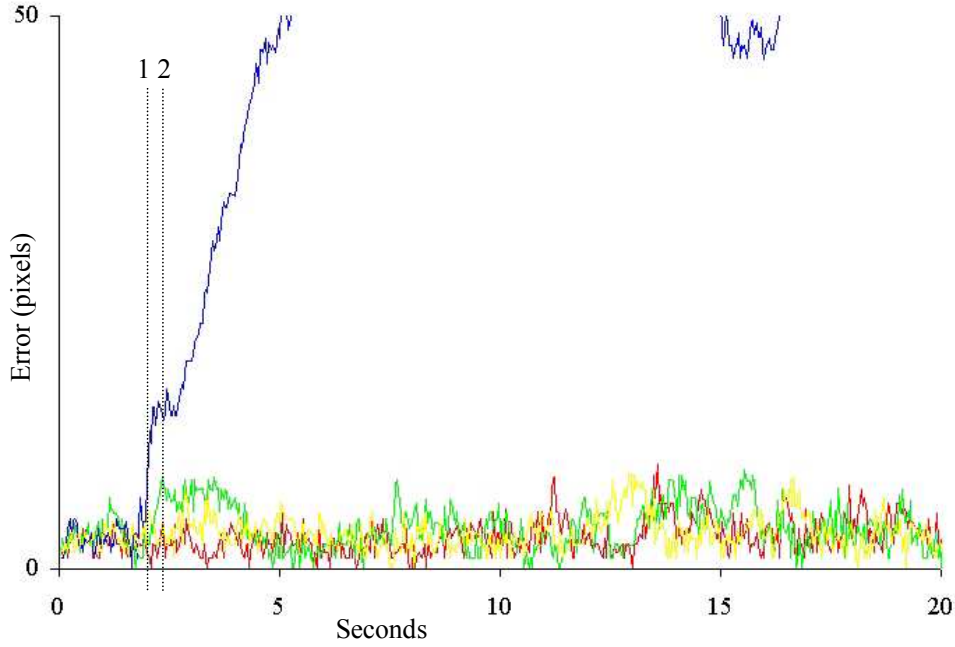


Figure 4.34 Residuals of groundtruth and actual data for sequence 1, with interesting frames marked with vertical bars. The bars represent (left to right): (1) Blue loses target (2) Green jostled off position when blue hijacks target.

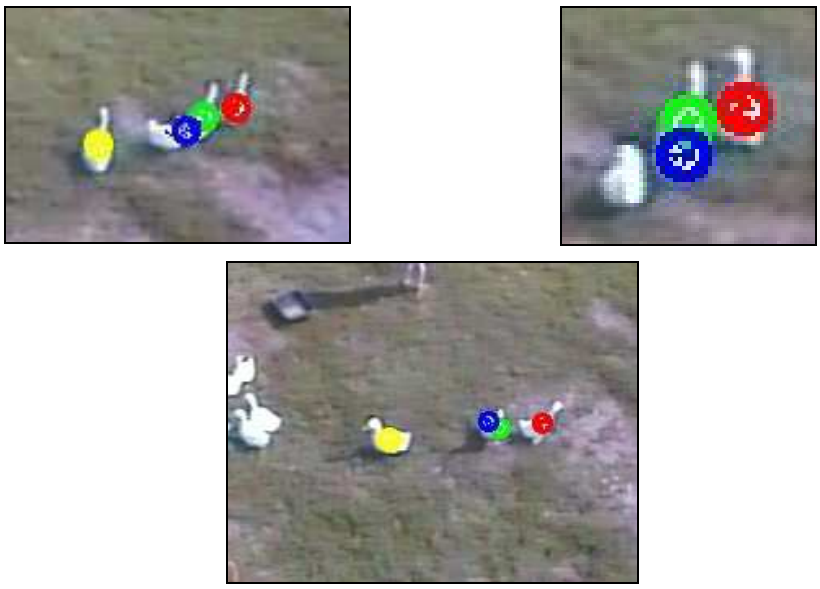


Figure 4.35 Output frames relating to the time markers in Figure 4.34. Clockwise from top left: (1) Blue loses target (2) Green jostled off position when blue hijacks target. The final image represents the final state of the tracking on the last frame, with only the blue tracker off-target.

It can be seen from Figure 4.32 and Figure 4.33 that the resultant tracks quite closely match the groundtruth, with the exception of the blue tracker which

moved onto the green target (Figure 4.35). This occurs in approximately the same place that the tracking of the blue target failed using the Condensation algorithm (Figure 4.20 shows the Condensation residuals and Figure 4.34 show the MCMC with interaction residuals.) Three targets have their identities maintained throughout the sequence, compared with two for Condensation. Additionally, the two that *did* have their identities maintained using Condensation did in fact lose tracking temporarily for a section of this previous sequence (see points (2) and (4) on Figure 4.20.) This did not happen with the MCMC MRF algorithm.

The RMS errors for the red, green and yellow tracking estimates are 3.3, 4.2 and 3.6 pixels respectively. These values are below their equivalents for Condensation during this sequence (for which red produced and RMS of 4.6 pixels and yellow 3.8 pixels) suggesting this MCMC algorithm may be more accurate, though the increase in accuracy is only slight.

4.10.2. Sequence 2 results using MCMC MRF

Figure 4.36 Groundtruth paths for sequence 2

Figure 4.37 Result paths for sequence 2

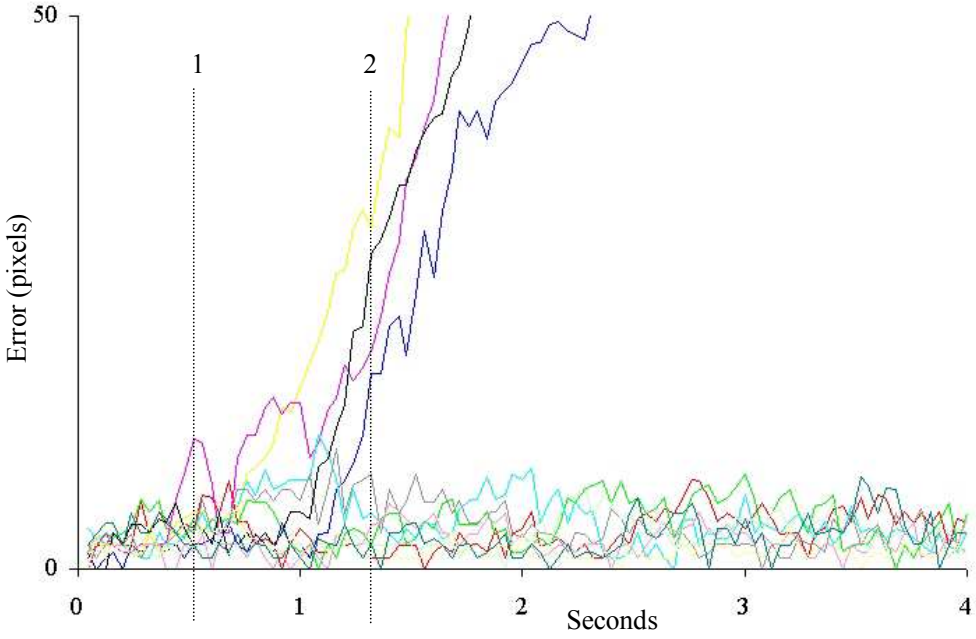


Figure 4.38 Residuals of groundtruth and actual data for sequence 2, with interesting frames marked with vertical bars. The bars represent (left to right): (1), the pink tracker has temporarily lost the target due to the rapid acceleration of the target. After a number of frames on the fringe of successful tracking, at (1) the pink target is lost irrecoverably.

Figure 4.39 Output frames relating to the time markers in Figure 4.38. From left to right: (1) Pink beginning to lose target (2) Pink has lost the target irrecoverably.

From the residuals graph (Figure 4.38) tracking seems comparable to condensation (Figure 4.24): 4 targets are lost throughout the sequence. However, note that with the targets for which tracking is not lost, the residual errors appear lower and more stable. The reason for the high number of targets being lost using both algorithms may be that the sequence involves large accelerations which push the dynamic models of both tracking algorithms to the limit. Indeed the 4 targets for which tracking fails are lost when the fast flocking motion starts: this motion

is as if the ducks have been startled by something, and is impulsive in nature, as the velocity rapidly rises from near-zero to a maximal speed for the ducks.

Note how the dark pink tracker temporarily loses tracking at 0.5 seconds (marker 1, Figure 4.38). This is due to the rapid acceleration of the animal and is a repeat of the situation that occurs using Condensation (see Figure 4.24). The tracker is unable to completely regain the target and loses it irrecoverably at about 1.3 seconds (marker 2).

4.10.3. Sequence 3 results using MCMC MRF

Figure 4.40 Groundtruth paths for sequence 3

Figure 4.41 Result paths for sequence 3

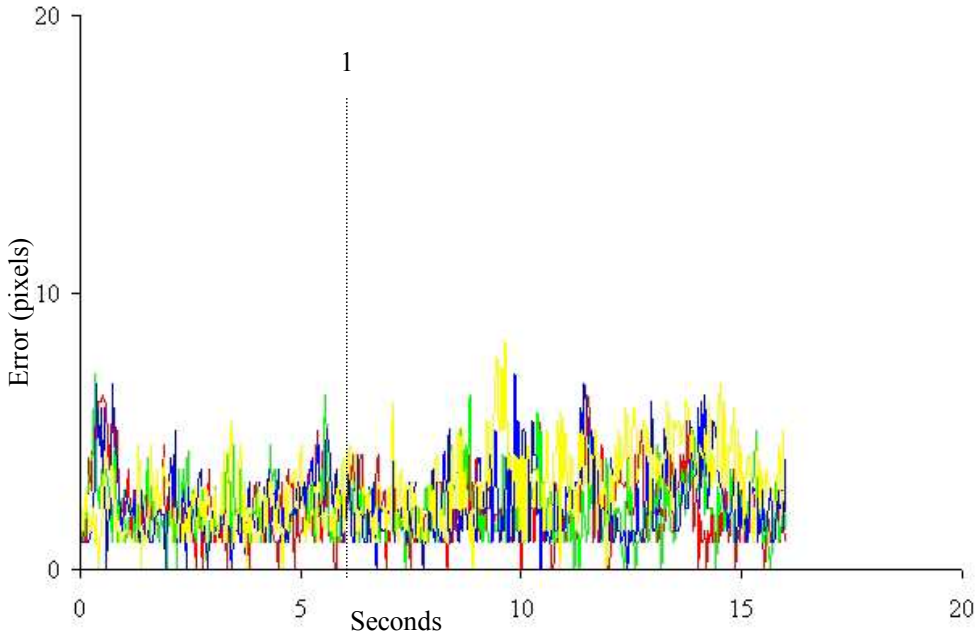


Figure 4.42 Residuals of groundtruth and actual data for sequence 3, with interesting frames marked with vertical bars. This bar (1) represents a point at which the Condensation trackers produced ambiguous estimates of the red, green and yellow targets, but MCMC MRF produces much more accurate results.

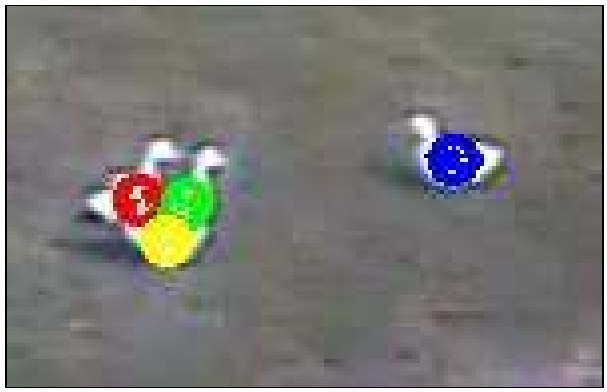


Figure 4.43 Output frame relating to marker (1) in Figure 4.42. Compared to the results of the Condensation algorithm at the same point in this sequence Figure 4.29 (top frame), all the targets are well placed and unambiguous (the samples are not split across multiple targets).

As can be seen both from the residuals in Figure 4.42 and from manual observation, this tracker performs very well on this sequence of two pairs of ducks approaching each other and crossing paths. The interaction function performs

well at preventing the tracking becoming confused when targets are in close proximity (Figure 4.43), producing a very good tracking result.

The RMS values for the four tracking estimates are 2.5, 2.5, 2.7 and 3.3 pixels respectively, which, given a baseline noise of about 1 pixel for the groundtruth, is a very good result: all the errors are well within the bounds of a duck. The blue track RMS is slightly less accurate than was recorded for Condensation (2.7 pixels versus 2.5 pixels), though this is only a small difference.

4.10.4. Sequence 4 results using MCMC MRF

Again, the feeding sequence was tested, this time with the MCMC MRF algorithm.

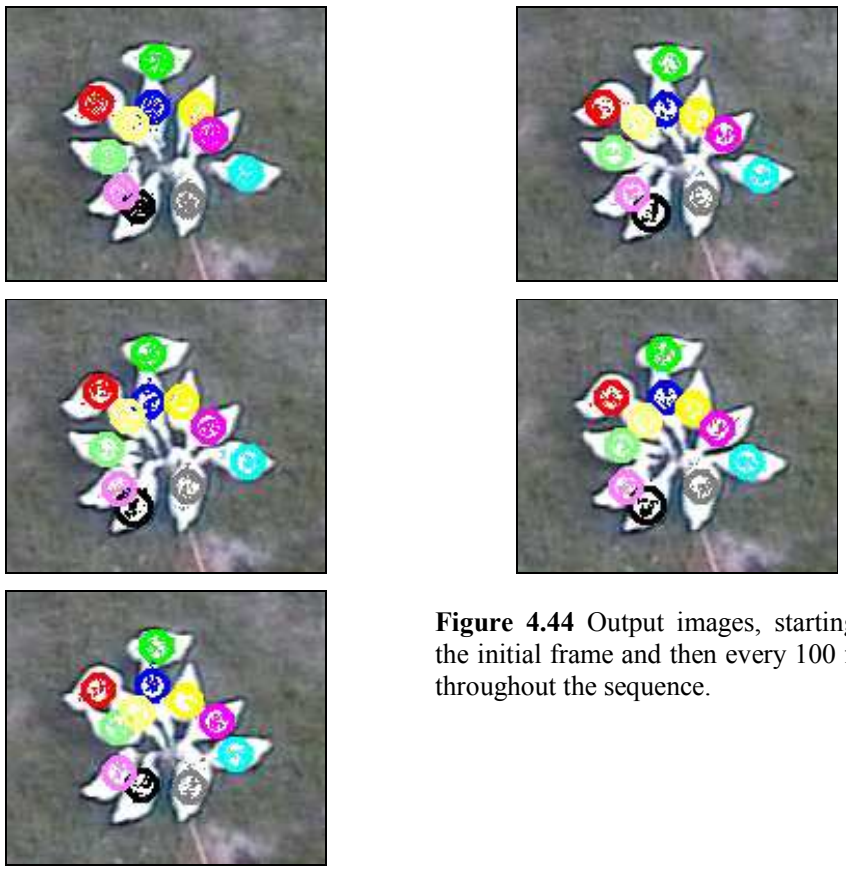


Figure 4.44 Output images, starting with the initial frame and then every 100 frames throughout the sequence.

Note how target identity is maintained, despite the targets being very close to each other throughout the whole sequence. This clearly demonstrates the power of the interaction-handling motion model in preventing the tracking estimates from moving to targets which are already being tracked.

4.11. Discussion

Using residual graphs as a measure of success for the tracking proved a useful visual aid, however it was found that they must be used alongside qualitative manual observation of the sequences. If a tracker jumps onto a target which stays in close proximity to the original target, then the RMS may remain low even though there is clearly a tracking error present. This problem is illustrated in section 4.8.6.

The image of the green and blue tracking estimates sharing a common target (Figure 4.35 (2)) suggests that the circle model for measurements may not be the most appropriate. If the model fitted more closely the target's geometry, this kind of target-sharing should not happen, as there should be fewer viable locations for the tracker per target.

One further note on the experiments themselves is regarding the number of samples used. The sample count was kept constant across the algorithms on a per sequence basis. However, due to its consideration of the complete joint state in each sample, the MCMC MRF algorithm should perhaps have its sample count increased proportionately to account for this. However, tracking was still improved over Condensation despite this, and the processing time would have been unnecessarily increased (perhaps to much greater than that necessary for real-time) by using increased numbers of samples. The sample count was initially kept constant per number of targets in the theory that the algorithms would run in comparable time. However, the multiple independent Condensation trackers were found to run generally faster (about 0.03 seconds per frame) than the MCMC MRF algorithm (ranging from 0.08 to 0.7 seconds per frame depending on number of targets). This increase in processing time for the MCMC MRF algorithm may be due to differences between the algorithms, such as the need to compute the interaction function. It should be noted that these times were recorded on a 1.4GHz P4, and no special effort was made with regard to optimization. On modern equipment, optimized to process from a live camera

stream, it is feasible that both algorithms could run in real time (0.04 seconds per frame) for all the target numbers used in this work.

4.11.1. Conclusions

As a comparison between the Condensation and Markov chain Monte Carlo MRF algorithms, these experiments highlight a number of key conclusions. It is clear that using multiple independent Condensation algorithms is not a successful solution to the problem of tracking multiple similar targets. The resulting 'best target hijacking' problem is illustrated in all the Condensation experiments, but is particularly noticeable in sections 4.8.5, 4.8.7 and 4.8.8. Despite the sequences not being subject to any exceptionally complex dynamics, the tracking estimates still jump across to the targets providing the best measurement responses. Conversely, the MCMC MRF algorithm is much more capable of overcoming these problems thanks to its use of the interaction model. The success of this model is particularly apparent in sections 4.10.3 and 4.10.4, where complete tracking is maintained throughout the sequences.

In terms of accuracy, the MCMC MRF algorithm performs better than Condensation some of the time (e.g. Condensation's 4.2 pixels average RMS error versus MCMC MRF's 3.7 pixel average RMS error across successfully tracked targets for sequence 1), and slightly worse some of the time (e.g. Condensation's 2.5 pixel RMS error versus MCMC MRF's 2.7 pixel RMS error for the blue tracker in sequence 3). Neither of these differences are large, and may be attributable to noise. One possible explanation for the *decrease* in accuracy of the MCMC MRF algorithm is that when two targets are within the interaction function distance, the particles may suffer from a 'repelling' effect, causing samples slightly off target centre and away form the opposing target to be accepted with perhaps higher probability than samples dead centre but nearer the other target. Any *increase* in accuracy could be due to the MCMC sampling process allowing the state space to be more efficiently represented, providing in turn better accuracy estimates of location. This effect might be more noticeable when fewer samples are available to represent the space. Both these effects

should be explored in future research into the accuracy of the MCMC MRF algorithm; however, both effects also appear to be small and so will not be further investigated for this thesis, which is more concerned with robust tracking than precise tracking accuracy.

The power of tracking techniques which allow for multiple hypotheses is illustrated in 4.8.7. The yellow tracker is temporarily distracted by a proximate target, but regains tracking after a short while as the velocity differential between the two targets helps to discriminate them, and the more viable, correct hypothesis is allowed to dominate. A single hypothesis tracker (such as Kalman filtering) would most likely lose track irrecoverably at this point. Condensation is often able to maintain tracking in such situations where the velocity differential is such that the targets can be disambiguated using this information, e.g. two targets moving in different directions, as can be seen in 4.8.7: the irrecoverable target hijacking error only occurs in the latter section of the video, where the ducks are nearly stationary.

Sequence 2, which featured fast flocking motions emanating from an initially stationary group of ducks proved a very hard sequence to track for both tracking methods. This appeared to be because of the large acceleration changes of the animals. There are two solutions which should improve tracking in this situation. The first is to tune the parameters to allow the trackers to be more likely to maintain tracking through the accelerations. Second is to use global motion information about the group's movement to help keep the tracking estimates on target, as the targets move in a similar fashion to each other throughout the fast motions, and so may be able to keep each other on track. The second of these is particularly interesting as it suggests that an algorithm which could somehow make use of social motion information may be able to help tracking. This theory will be tested in Chapter 5.

Overall, the MCMC MRF algorithm can be considered an improvement on traditional Condensation trackers, especially when the targets are moving slowly and in close proximity. This is certainly true of the robustness of the tracking, and

sometimes true of the accuracy, though the accuracy does also decrease by a small amount sometimes. The ability to maintain tracking of individuals in close proximity is due to the use of a joint state space, where inter-target interactions can be taken into account. The MCMC MRF algorithm, then, will be used as the benchmark from which to develop and test a new socially-aware algorithm, presented in Chapter 5.

Chapter 5: Using social information to improve tracking performance of groups

5.1. Aim

It has been shown in the previous chapter that the MCMC joint state tracker with an interaction function is a suitable tracking algorithm for multiple targets, but still does not produce perfect tracking in all situations. This chapter will examine the hypothesis that *social motion* information can be incorporated into a particle filtering tracking framework to make the tracking of groups of targets moving in a related way more robust and accurate.

5.2. Motivation

Multiple animals together often means more than just a collection of individuals. Often groups form within the collection of individuals; such groups may be concerned with mating, foraging, protection or some other task or behaviour. How a group is defined can be a subjective judgment in itself, often dependant on the tracking domain. Often, spatially close targets are considered a group (McKenna et al. 2000a; McKenna et al. 2000b; Cupillard et al. 2001), although this is not necessarily a good choice generally. This thesis will test another method based on similarity of motion as a measure of group identity.

Individuals may perform joint actions, such as walking to a food source and eating together, which add a *social* aspect to the motion of the group. Two animals

might follow one another around preparing to mate or looking for food. Insect swarming and fish schooling are examples of social events which contain hundreds or thousands of individuals. The behaviour of such large-scale groups has been modelled, typically using physical forces between the individuals in the models (Okubo 1986; Reynolds 1987). Similarly, crowds of people in public places contain many individuals with some global motion imposed upon them. The actual social effects causing these motions and joint behaviour are difficult to quantify, and often it is only the resulting 'look' of a group behaviour that is modelled; the reason *why* the flocks behave as they do is unexplained. The flock or school is driven by potentially complex interactions which can only be hypothesised. The level of complexity of these rules is still under debate (Viscido et al. 2002). Group motion models are still being hypothesised and theories are still being formed about how the motion of a group of animals is determined when, for example, no one member knows which individuals have the most salient direction information (Couzin et al. 2005). The ongoing and varied work on group motion models means that any new method developed in this work would benefit from being general in nature and not tied to any one particular method – there is no agreement yet on one high-level group behaviour model, so to use one particular method might over-specialise the model.

To understand what social events actually are, it is necessary to understand the targets being tracked. Tracking 'live' targets, such as animals or people, is typically a harder task than tracking inanimate targets, as their motions are the result of a hidden layer of internal reasoning. Inanimate objects however are governed by external forces. Snooker balls for example, follow clearly defined rules of physics and these predictive models can be incorporated into tracking algorithms to increase robustness. For example, if a ball is dropped, using a constant acceleration model such as is imposed by gravity is sensible (Isard and Blake 1998a). With targets that are alive, such physical models are still applicable (a person free-falling will be accelerated by gravity) but the target may impose their own unpredictable constraints on their motion (opening a parachute, for example). With more than one such target, this effect is compounded. When multiple, living targets are present they are able to interact with each other, their

motions changing based on the state of their colleagues. Such interactions interfere with the predictions made by standard linear motion models: one minute a duck is happily walking along a straight trajectory, the next he decides to take a detour to walk up to another duck. They may circle around each other or even mate – such interactions are very difficult to model because two similar targets are moving in close proximity along complex trajectories determined by internal goals. Similarly hard problems are encountered when tracking multiple people. They may move about seemingly at random, stopping to chat with one person and trying to avoid several others.

However, despite their differences, both types of targets may produce what can be considered as coordinated behaviour. Dropping a handful of snooker balls off a tall building (not that this author would recommend such an action) produces a group of targets whose motion is such that they maintain their coherence as a 'group'. If one target becomes obscured, this ball's position and motion can still be predicted using information about how the other balls move. The same is true of a group of animals or people who, rather than being driven by the laws of physics are driven by a set of *social motion rules*, such as escaping a predator or looking for food. Obviously, subtle divisions of social rules probably exist, but determining what they are can be problematic. The point of this is that the motion of groups of targets, be they social or otherwise, are often governed by higher level rules which, although not necessarily explicitly known, do govern the global behaviour in a predictable way. Using this theory it is conceivable that an algorithm could be developed which uses knowledge about *social motion* to guide the tracking of a group of targets, no matter what the targets are, as long as they exhibit some sort of coordinated motion.

5.3. The problem of tracking social groups

Most, if not all existing group tracking algorithms decouple the tracking itself from the social aspect of the grouping, flocking or herding motion (depending on the type of target!). Even if the final aim of an algorithm is to be able to label the behaviour of the observed targets, such as antisocial behaviour (Cupillard et al. 2001) or events in a car park (Ivanov and Bobick 1999) etc., the final 'behaviour labelling' is rarely used to actually guide the tracking itself. One algorithm which does combine behaviour labelling with the motion model in use is mixed-state particle filtering (Isard and Blake 1998a). Here the labelling of the motion happens implicitly based on the dominating motion model; an example use is the recognition of temporal gesture trajectories (Black and Jepson 1998). Variations on this technique include using multiple cyclic hidden Markov models to model healthy or lame motions for particles used to track a cow, both to apply a suitable tracking model and assign a behavioural label to the target (Magee and Boyle 2002). The dominant model that best fits the data will lead to more samples being associated to this model over time. A simple classification can then be made by observing the number of samples that represent each model

It has been shown in the previous chapter that social interactions between targets can degrade the performance of tracking algorithms. Most data association methods make the assumption of target independence; that is, when two targets' paths cross, their motion continues just as before they came together. With social animals and people, this requirement is often not met: animals meet up and interact for some unpredictable amount of time, and then continue, maybe on the same trajectory as previously and maybe not. It was shown in Chapter 4 how simple spatial interaction models can be used to improve tracking in these situations (Khan et al. 2004). These spatial models, however, make no assumptions about how the targets are *moving*, simply what to do then they are close to one another. Existing tracking algorithms should be able to track the members of a group as long as:

1. The dynamics of the group fall within the dynamics covered by the tracker
2. Members of the group are not fully occluded

Problems arise, however, were these conditions are breached. With the first case above, increasing the process noise of the tracker might allow it to capture more extreme motions, but it may also lead to increased chance of the capture of similar looking clutter – a problem compounded by the similar targets present in group situations. It would therefore be advantageous to track the group with the smallest amount of process noise possible.

With the second case, when targets become completely occluded, it is all the tracking algorithm can do to continue the prediction of position based on the last known dynamics of the target. How long this 'ghosting' continues for and whether it gets caught on clutter during this stage depends on the implementation of the algorithm and the properties of the occluding clutter - whether it is camouflage (producing uniform measurements) or distracting (producing measurement peaks) in nature. On occasion, the obscuring clutter so closely fits the colour model of the desired target that the momentum of the recently hidden target's tracker is maintained via positive feedback from the clutter measurements. Therefore, the tracker continues on its set course, and is likely to be in the correct place to recapture the occluded target when it emerges on the other side (assuming the target continues its trajectory). If the clutter is largely different in colour from the target, the algorithm receives few positive measurement scores. If there is a small area of the clutter that scores more highly than the rest, this area will act as a magnet to the floundering particles. Particles falling over this area will be likely to be propagated to the next time step, causing the velocity to tend towards zero (if the clutter is stationary), and the tracking of the target to be lost, as in Figure 5.1.

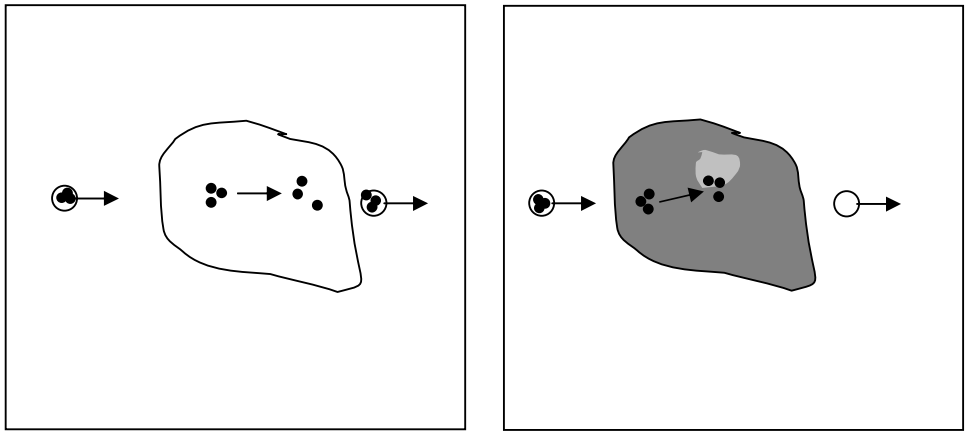


Figure 5.1 Illustration of how clutter colour can affect whether a target (white circle) is successfully tracked. Black circles represent approximate particle cloud location. In the left scenario, the colour properties of the clutter are similar to that of the target; the momentum of the particles is upheld by high scoring measurements and the particles meet the target upon leaving the clutter. In the right scenario, the clutter does not match the colour model of the target. Therefore, *relatively* high scoring patches within the clutter become very attractive to the particles, and tracking is lost.

If the target is occluded by something which produces a uniform measurement response (camouflage), then the area of the state space explored by the tracker increases. This can happen when the confidence level falls when using a Kalman filter or when a particle filter is presented with a situation in which all particles have equally scoring observation measures, and an example of this situation with a particle filter is presented in Figure 5.2 :

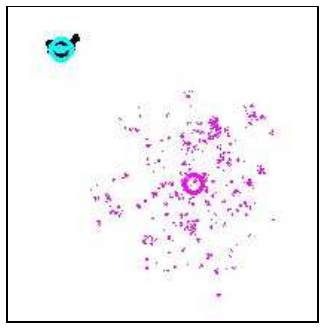


Figure 5.2 Example of particle spread when no target observations exist for a particle filter. Note how constrained the particles are when measurements exist that fit the observation model (*top left*) and how the particle area increases over a few frames when no reinforcing measurements exist.

Although this may manage to capture the target when it reappears, it only does so by 'shooting in the dark' – it is just as likely to settle upon background clutter that generates a high enough measurement for one of the particles. Certainly, if the target changes velocity during occlusion, the tracker has little hope of recapturing the target.

Despite their interaction function, Khan et al.'s MCMC MRF algorithm still falls foul to the above effects (again, as in the previous chapter, MCMC MRF refers to Khan et al.'s Markov chain Monte Carlo particle filter with the Markov random field interaction function (Khan et al. 2004)).

5.3.1. A general solution to these problems using social knowledge

The key to overcoming these problems is doing what a human observer might do: incorporating more information. When concentrating on one target, by definition all information about its motion disappears when the target is occluded from view, or camouflaged. However, when a group of social targets are studied together, often patterns can be seen in group movement. These patterns could be used to aid predictions of their motions.

This work tests the hypothesis that *group social motion* effects can be used to aid tracking by allowing the motion model of one target to use information about how one or more previously coordinated targets are moving. For example, knowing that two targets are behaving in a similar way may allow us to hypothesise the motion of one occluded target based on the motion of the visible target. Imagine a group of people walking through a park, and after a time a small number of members of the group are obscured by trees and bushes in the foreground. It is possible to make judgements about where they are and where they will re-appear by looking at the position and motion of the *visible* members of the group. Indeed, we as people are likely to make such judgements without conscious effort. It is logical to use a similar technique to improve the tracking of groups of targets exhibiting such behaviour. These *temporal* social interactions are a completely different social event to that of the *spatial* interactions modelled by Khan et al., but can and should exist alongside such a method of keeping multiple tracking

estimates from all following one target. It is sensible to include this previous work in the final algorithm, as it helps resolve the target-hijacking problem that is common with interacting targets, whereas the new social motion algorithm will also model dynamics using the motion of coordinated targets as a guide. However, the new temporal social interactions work is independent of these spatial interactions, and so the *spatial* interaction model of Khan et al. can be omitted if required.

Using temporal social interactions should improve upon results for the first problem listed in section 5.3, where the dynamics of the tracker's motion model are on the verge of losing the targets. Using global group motion should improve the tracking of difficult sequences where the dynamics of the tracker are stretched to the limit (such as Sequence 2 in Chapter 4) with lower process noise than traditional methods which do not include social motion components. If, as in this example, a group of ducks suddenly exhibits a fast moving flocking motion, if any of the tracking estimations falls behind and loses the target (such as the pink tracker in experiment 4.10.2 in the previous chapter), then it is hypothesised that using information about how the other members of the flock are moving will help to keep the tracker on target.

The occlusion problem could be rectified using information about how the whole social group is moving to predict the motion of the occluded target belonging to this group. This would help prevent the tracker getting stuck on relatively high-scoring clutter, and even guide the tracker through an occluded target's velocity change if this was reflected in the motion of the other members of the group. It will also prevent the particles from spreading out excessively in the absence of a reinforcing measurement.

In addition, the general accuracy of tracking may be improved as well. Target location noise often arises because of short term deviations in the target's motion as compared to that modelled by the tracking algorithm. Such deviations may cause the algorithm to alter its estimation for the next timestep, thus compounding the error. Taking into account group motion when tracking individuals leads to

such erratic motions being filtered out in a way consistent with the motion of the group as a whole. This should in turn help alleviate some of the problems of localised, independent noise on the targets, and so increase the general accuracy of the algorithm.

By improving tracking results for these problematic occasions, a new, powerful technique for tracking multiple social targets should emerge. This new technique can make use of the mixed-state particle filter, using *social group motion* metrics to assign motion states, social or otherwise, to the tracker. Therefore, this work introduces a novel method to improve the tracking of groups of social targets, by incorporating *social* motion information into a mixed-state-derived particle filtering algorithm.

5.4. Algorithm development

The success of Khan et al.'s methods (Khan et al. 2004) lies in the ability of the tracker to be aware of where the targets are in relation to each other. When the targets are close to one another, the motion model is able to suppress particles that drift too close to nearby targets. Thus the problem of best-target hijacking is overcome. Khan et al.'s algorithm, however, does not make use of any information about how the targets are *moving* in relation to each other. It models social effects in space (i.e. proximity to other targets), but not *time* (how the targets are moving together). It is hypothesised that an extension incorporating time-based social information into the process density, when tracking groups of targets moving in any sort of coordinated fashion, would increase the robustness and accuracy of the tracking. Using motion history information for individual targets (Magee and Boyle 2000) has been shown to improve speed and robustness in the past, and so it is reasonable to assume that including group motion history information will yield similar benefits. The description of the development of such a technique shall be presented next, followed by the experimental evaluation.

5.4.1. A socially aware process density

The new Motion Parameter Sharing (MPS) algorithm needs to have two basic components to its motion model: an 'internal' motion representing the target's motion in the absence of social factors, and a 'social' component which is affected

by the motion of the other targets in the same social group. The foundation of the new MPS algorithm was drawn from two important but disparate existing algorithms. As was seen in the last chapter, the MCMC MRF algorithm (Khan et al. 2004) allowed interacting behaviour between targets to be modelled and this helped prevent the target hijacking problem that was common when animals and insects interact. This was a spatial model which essentially suppressed measurement values as targets became nearer to each other. As it does not depend on the previous state, this interaction term, ψ, can be used to simply modify the acceptance ratio of the MCMC algorithm:

$$a_s = \min\left(1, \frac{P(Z_{it} \mid X'_{it}) \prod_{j, j \neq i} \psi(X'_{it}, X_{jt})}{P(Z_{it} \mid X_{it}) \prod_{j, j \neq i} \psi(X_{it}, X_{jt})}\right) \tag{8}$$

where Z_t is the measurement at time t, i is the target being moved, j represents the other targets in the group, X_t is a sample statespace for time t and X'_t is a proposed new sample statespace.

Note that this leaves the motion model itself, $P(X'_{it} \mid X_{i(t-1)})$ unchanged. So, this algorithm has the advantage of a spatial interaction model which incorporates knowledge about a target's neighbours, but the actual dynamical model does not take advantage of this knowledge: the samples are affected by the interaction *after* movement.

The second algorithm, mixed-state Condensation (Isard and Blake 1998a), does not implement a joint tracker and hence it has no awareness of what other targets might be doing. What it does offer, however, is the ability to automatically select between multiple available motion models by means of an extended state, X:

$$X = (x, y) \tag{9}$$

where x is a statespace describing the target's parameters, and y is a discrete variable labelling the current motion model in use. The process density then becomes:

$$P(X_t \mid X_{t-1}) = P(x_t \mid y_t, X_{t-1})P(y_t \mid X_{t-1}) \quad (10)$$

The transition between motion model states is modelled as follows:

$$P(y_t \mid X_{t-1}) : P(y_t = j \mid x_{t-1}, y_{t-1} = i) = T_{ij}(x_{t-1}) \quad (11)$$

i.e. the evolution of y is governed by a state transition probability matrix, T_{ij}, where i and j are states in y.

It is proposed that the new MPS algorithm will have to use an MCMC sampling methodology because of the necessity to include a joint statespace. Additionally, as the targets are interacting, it is sensible to include the spatial interaction function of the MCMC MRF algorithm. The motion model will then be extended to incorporate the mixed state model (from equation (10)) to allow for different motion parameters to include social motion information when present. This social motion information will be in the form of motion parameters shared from socially coordinated targets. However, instead of using a pre-determined matrix of transition probabilities to determine the state (equation (11)), it is proposed that the *motion of the targets themselves* can be used to determine whether the state should relate to individual motion or social motion. This will be done by looking at how well the targets' motions are correlated over time, thereby introducing a new *temporal,* socially-aware element to the process density.

5.4.2. Defining the groups

The motion parameters should only be shared to targets belonging to the same 'social' group i.e. targets which are likely to move in the same way. There are many possible ways of defining such groups. Proximity is one measure which has been used before to model social interactions (Khan et al. 2003), but being in close proximity does not mean the targets are necessarily moving in the same way. This behaviour was noted in one of the duck videos, where the motion of two ducks was found to be highly correlated (see Figure 5.3) over a period of over

2 minutes (3,800 frames), despite them becoming separated at various times throughout the sequence, sometimes by a large distance.

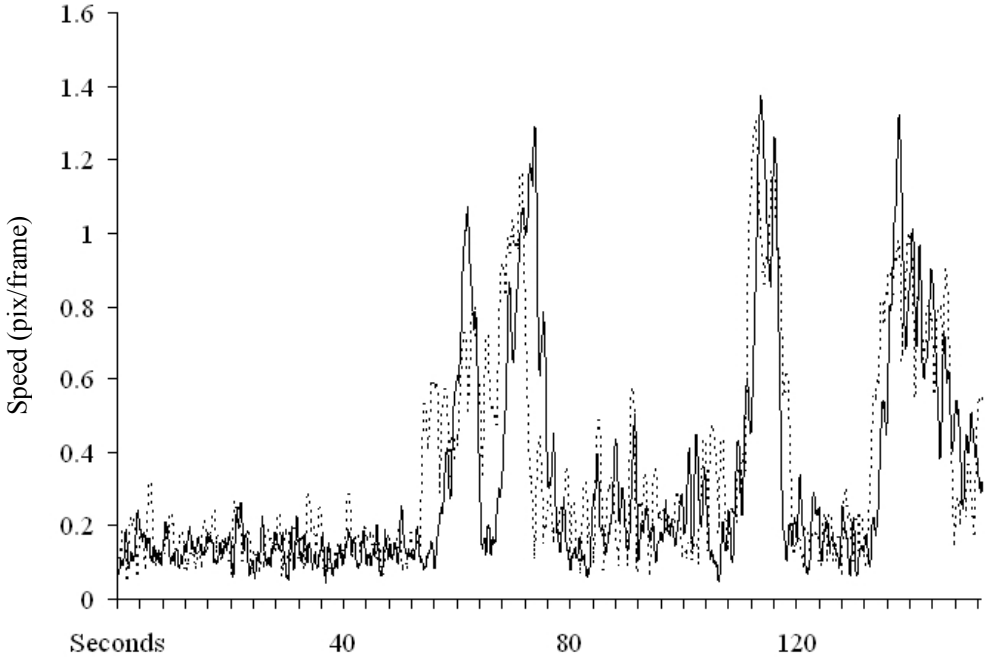


Figure 5.3 Speeds of ducks t5 and t6 (r^2=0.58)

This example illustrates that although distance is sometimes used as a metric to divide targets into groups, this is not the best strategy for this work. A better measure of allocating targets to groups is to use how the targets move in relation to each other. To quantise such a parameter, correlations can be calculated over the targets' motions over time. For example, the targets' velocities, speeds, path curvature, path noise etc. could be correlated over time, with targets behaving in a similar way (as determined by this parameter) being classed as a 'social group'. Targets with a higher correlation value are more likely to be moving in the same way over the parameter for which correlation was calculated.

For this work, speed was chosen as the parameter on which to calculate correlations. Figure 5.3 shows that speed should be a meaningful correlation metric. Correlating *velocity* has advantages and disadvantages over speed, the main difference being targets would have to be moving in the same direction to be

considered a group. The consequences of this will be examined in future work, but initially speed was used as this was thought to allow for more generic social situations, and has been shown to be a parameter over which social pairs of ducks are correlated. A distance measure was omitted from this work as the social pair of ducks in Figure 5.3 were sometimes separated by large distances. It should be noted though that the parameters used to determine the groups can be different depending on the situation under study, and can also be different from the parameters shared in the motion stage of the algorithm.

5.4.3. The MPS algorithm

The first part of the algorithm needs to incorporate a joint motion history in the form of correlated speeds, as follows. Every time step, a correlation matrix between all the targets' speeds is calculated using Pearson product-moment correlation coefficients across a sliding window of the past N frames. The value N should reflect the scale of the motions. This is a complicated issue and deciding on N requires future investigation, but for this work a value in the order of a second was used. This matrix will therefore store how well the targets' motions are correlated over the previous time window.

To be correct, each individual particle should calculate the correlation matrix based on its own internal joint motion history for the past N frames. However, this is computationally expensive for two reasons: first, each particle must maintain a joint history of the past N frames, which produces a large particle size (the temporal Markov chain constraint introduced by e.g. (Isard and Blake 1998b) was to prevent the need for this). Second, the processing time for calculating the correlations within each particle would be large. Therefore, a practical approximation is necessary to make the algorithm run with limited resources. To do this, one set of correlations is calculated each timestep across the past N frames using the best-estimate speeds of all the targets. The speeds are smoothed over a small window (about $1/6^{th}$ second) to eliminate erratic changes prior to correlation calculation. These speeds can be adjusted for position on the ground plane using a similar method to the radius setting model in Chapter 4.

When a target's statespace is propagated forward, the motion model can either use the target's own parameters or it can use the motion parameters of a correlated target. For each particle, a random third-party target is selected. The motion parameters from the third-party target may be used for the current particle with a probability proportional to the correlation coefficient between the two, with a practical lower threshold set to eliminate noise called the correlation threshold. This correlation threshold determines how tightly animals' motions must be correlated in order to be considered a 'social group'. If the correlation coefficient is low, the particle is likely to use its own internal parameters. So, for each target a pairwise relationship is assessed with a randomly chosen target. The first target's tracking representation may or may not use this other target's motion parameters with a probability related to their level of correlation. Across the complete set of particles, this collection of pairwise relationships represents a prediction of motion taking into account the whole group (or groups) of socially coordinated targets.

The motion model parameters used are therefore determined by the level of correlation between the targets' speeds over an elapsed time window. The process density is then a modified form of that used in mixed state Condensation (Isard and Blake 1998a):

$$P(X_t \mid \chi_{t-1}) = P(x_t \mid y_t, \chi_{t-1}) P(y_t \mid \chi_{t-1}) \qquad (12)$$

where X_t is the state at time t including y, x_t is the state at time t excluding y, and y_t is a discrete variable labelling which target's motion history should be used to process the target's state forward at time t, ie. using its own motion parameters, or those shared from another target. χ represents the complete history of all information stored in X. This is a modified form of equation (10) incorporating the complete state histories required to determine which targets have been moving in similar ways. Isard and Blake (Isard and Blake 1998a; Isard and Blake 1998b)

make the assumption that this can be approximated by a temporal Markov chain, simplifying the process density to that in equation (13).

$$P(X_t \mid \chi_{t-1}) \approx P(x_t \mid y_t, X_{t-1}) P(y_t \mid X_{t-1}) \tag{13}$$

The Motion Parameter Sharing algorithm also makes the Markovian assumption for the *motion* at time *t*, as in the mixed-state tracker (Isard and Blake 1998a), but the state label y_t remains dependant on χ, i.e. it is dependant on state information with a history reaching further back than just the previous timestep, as is required to incorporate correlation information. In practice, χ is approximated by a limited history χ' (the correlation window) to alleviate some processing requirements. This leads to a process density of the form:

$$P(X_t \mid \chi_{t-1}) \approx P(x_t \mid y_t, X_{t-1}) P(y_t \mid \chi'_{t-1}) \tag{14}$$

This allows the use of a motion model dependant only on the last time step, and a discrete motion parameter label variable y_t dependant on some history function – in this case correlation over a time window. The actual probability of target A borrowing motion parameters from target B in the processing forward of one sample, in a scene with N targets, can be practically calculated as follows:

$$P(y_{Bt} \mid \chi'_{t-1}) = \frac{1}{N-1} P(r_{AB}) \tag{15}$$

where $P(r_{AB})$ is the probability that the two ducks have been exhibiting correlated motion – for this work, this is considered equal to the correlation coefficient between targets A and B calculated over the sliding time window. For practical purposes, if $P(r_{AB})$ is below a fixed noise threshold, then $P(r_{AB}) = 0$; i.e. if there is only a weak correlation between the targets, they are considered to be not correlated at all.

The actual probability of a target using its own motion parameters in a sample process can be calculated using equation (16).

$$P(y_{At} \mid \chi'_{t-1}) = 1 - \sum_{N}^{N \neq A} P(y_{Nt} \mid \chi'_{t-1}) \tag{16}$$

Together these probabilities form the equivalent of the transition matrix T_{ij} in equation (11).

One problem with the MCMC algorithm is that the acceptance of samples into the new set is calculated by selecting the relatively best scoring measurement out of two alternatives (steps 3b and 3c in Algorithm 5.1 below). This means that in the complete absence of a target, such as during occlusion, the tracker can quickly become attracted to relatively high scoring areas of clutter (see Figure 5.1 for more details), even though when measured absolutely these areas produce low scoring measurements. In the absence of any further guiding information, this is a common problem for such a tracker. However, with social information available we can use this new knowledge to guide the tracker instead. To overcome this issue, an additional acceptance condition was added to step 3c, whereby if the measurements are both below a practical threshold T_m (i.e. there is definitely no target present), and a correlated motion model has been used for this sample, then accept the sample anyway. This means that if a tracker is following an occluded target using motion information from coordinated colleagues, it will use this method to add samples to the new set, rather than allowing relatively high scoring, but ultimately poor, clutter measurements to disrupt the tracking. In practice, this step has little effect on the original algorithm during normal tracking. For example, after multiple tests on sequence 2 from Chapter 4, it was found that this acceptance method was only invoked 0.4% of the time versus the other acceptance methods, probably due to the lack of occlusion in this sequence. This special threshold is therefore only employed during the described occlusion-type event, and should not affect tracking where the target produces a measurement.

The main MPS-extension is presented in steps 3a(iii) and 3c below, using the semi-Markovian equation (equation (14)) for the process density, modified from Isard and Blake's mixed-state particle filter.

MOTION PARAMETER SHARING ALGORITHM

1. **State of targets** at *t*-1 represented by a set of samples $\{X_{t-1}^{(r)}\}_{r=1}^{N}$. Each sample contains information on the complete joint state
2. **Initialize** the MCMC sampler at time *t* by drawing X_t from the predictive density
3. **Metropolis-Hastings iterations.** Obtain *M* samples from the posterior. Discard the first *B* samples for sampler burn in. In detail:
   a. **Proposal:**
      i. Randomly select a joint sample $X_{t-1}^{(r)}$ from the unweighted samples at *t*-1.
      ii. Randomly select target *i* from the *n* targets in this sample. This is the target to move.
      iii. Test whether to take motion parameters from a random other target, dependant on how well the targets are correlated. If this fails, use own motion parameters. Apply motion model to this target to obtain $X_{it}^{'}$.
   b. **Calculate acceptance ratio:**

$$a_s = \min\left(1, \frac{P(Z_{it} \mid X_{it}^{'}) \prod_{j, j \neq i} \psi(X_{it}^{'}, X_{jt})}{P(Z_{it} \mid X_{it}) \prod_{j, j \neq i} \psi(X_{it}, X_{jt})}\right)$$

   c. If $a_s = 1$ then accept the proposed sample $X_{it}^{'}$, i.e. set the *i*th target in X_t to $X_{it}^{'}$. Else if $a_s<1$ we accept the proposed sample with probability a_s. If rejected, leave *i*th target in X_t unchanged. Also, accept $X_{it}^{'}$ if the motion parameters have been shared from another target, and the measurement response is below a threshold, T_m.
   d. Add a copy of X_t to the new sample set.
4. $\{X_t^{(s)}\}_{s=1}^{M}$ represents the estimated joint posterior.

Algorithm 5.1. The MPS steps incorporated into a Markov chain Monte Carlo particle filter, with an additional interaction function (Khan et al. 2004).

So, using this new algorithm correlated targets are able to guide each other through clutter, occlusion and other problematic events. The algorithm will be tested in the following sections on various kinds of artificial and real life target sequences.

5.5. Experiments with path perturbation and occlusion

5.5.1. Introduction

The aim of these experiments is to assess the performance of the new MPS algorithm compared to the previous existing best as determined by the previous chapter of this thesis, i.e. the MCMC MRF algorithm. Some general problem areas where tracking was likely to fail using traditional techniques are:

1. During full occlusion, where the tracker becomes attracted to 'best' clutter in the absence of a target measurement
2. During path perturbation, i.e. high noise on a target's path

It is hypothesised that if the targets are exhibiting correlated motion, tracking accuracy should be improved in these situations. The first tests for the new MPS algorithm will assess its effectiveness in the above scenarios, as well as demonstrating the effect of the algorithm on sequences where neither of these special events occur, to enable a direct comparison between this MPS algorithm and the existing MCMC MRF tracker.

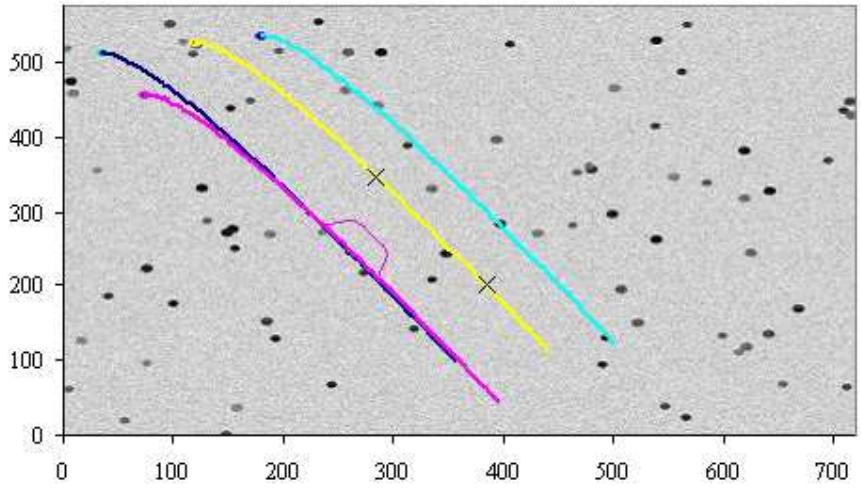


Figure 5.4 Diagram showing the paths of the four targets in the artificially generated test sequences. Note the perturbations on the pink path (used in Experiment 2). The crosses on the yellow path delineate the simulated occlusion phase for Experiment 3. The colours in these images have been inverted for clarity.

For these artificial sequences, the targets were white circles, and the measurement model favoured such an object. The parameters of the tracker were kept the same as for duck tracking in the previous chapter. Gaussian image noise ($\sigma = 4.0$) was added independently to each frame. Also present was distracting similar background clutter (see Figure 5.4). A correlation window, *N*, of 50 frames (two seconds) was used. The advantage of using artificial sequences is that the experimenter has control over the targets (not so with animals!) and also a true ground truth is known. Both tracking algorithms used the same number of particles unless otherwise stated. RMS errors were calculated across all four targets. RMS is not presented for runs where at least one target's tracking is lost entirely, as this is an unrecoverable tracking situation worthy of special note.

5.5.2. Experiment 1: Group motion along simple paths

In this experiment using an artificial sequence, a group of targets move together along a simple path which contains no occlusions or perturbations. This test is intended to provide a direct comparison of accuracy between the MCMC MRF algorithm and the MPS algorithm. Each algorithm was run five times on the sequence, with the same initialisation, and the results compared to the groundtruth. On every run both algorithms maintained tracking of the targets throughout the sequence.

RESULTS

The results of the tracking are presented in the following table of RMS errors.

Run number	MCMC MRF algorithm (pixels)	MPS algorithm (pixels)
1	0.69	0.57
2	0.71	0.63
3	0.67	0.62
4	0.61	0.68
5	0.69	0.62

Table 5.1 Table of RMS errors in the comparison between the MCMC MRF and MPS algorithms on the simple paths test sequence.

The means of RMS error from Table 5.1 above show that the MCMC MRF algorithm had an average error of 0.67 pixels and the MPS algorithm an average of 0.62 pixels. This reveals a slight increase in accuracy for the MPS algorithm ($t(8)=-2.04$, $p=0.08$). Run 4, however, did produce a small decrease in accuracy for the MPS algorithm. Closer inspection of this sequence does reveal, however, that these errors occur early in the sequence, *before* parameter sharing begins, while the correlation window is still being built. Therefore this error is not caused by the MPS algorithm, and could equally occur with the MCMC MRF tracker.

The processing time for the two algorithms is comparable, with both taking around 20ms per frame on an average specification PC (1.4GHz P4). Both algorithms are fast enough for real time application

DISCUSSION

The MPS algorithm has comparable, and usually slightly improved, accuracy compared to the MCMC MRF tracker in this social scenario. One reason for this improvement may be due to the way the estimated locations of the targets are calculated. The estimated location of a given target is given by the average of all the particle locations for that target, i.e. a best estimate of the probability density function. With MPS, the drift of each individual's probability density is constrained further by the common (and shared) group motion. Error across the whole particle set is therefore reduced, allowing more accurate prediction.

It should perhaps be noted than in a non-social scenario, the correlations would fall below the threshold and tracking would revert to standard MCMC MRF methods. It follows that the error in a non-social scenario would therefore be the same as for the MCMC MRF algorithm.

5.5.3. Experiment 2: Path perturbation

This experiment is designed to test the situation where one member of the group has to make a sudden and short-lived detour from its previous trajectory because, perhaps, of some obstruction in its path. The group of targets move together throughout the sequence, with the exception of the one target which undergoes the

path perturbation. Such a perturbation is essentially an example of localised but severe noise, and may distract traditional tracking methods by 'sling-shotting' the tracker off course, or by the tracker becoming attracted to clutter as it maintains the trajectory in the absence of target measures. In many situations it is desirable to not track the target throughout the perturbation, but to recapture the target when it rejoins its original path. This eliminates the perturbation 'noise' on the path. The implemented perturbation can be seen on the pink path in Figure 5.4

The following table shows the results of 10 runs of both trackers on this sequence with the path perturbation.

	MCMC MRF	MPS
Correct runs	2	10
Average RMS error for correct runs (pixels)	2.60	0.772

Table 5.2. Correct runs and RMS errors of correct runs (compared to the unperturbed groundtruth) for the perturbation sequence.

A 'correct run' is defined as one where the tracker maintains the identity of all the targets throughout the sequence i.e. at the end of the sequence they are positioned on the correct target. This is determined by manual observation. The RMS errors are only meaningful in situations where a run is correct, for reasons discussed in the previous chapter. Therefore, RMS errors are presented as an average across only the correct runs. As Table 5.2 shows, the new MPS algorithm was very successful, producing 10 correct runs out of 10. The MCMC MRF tracker was markedly less successful, managing to maintain tracking on only 2 runs out of 10 for the same sequence. There is a high level of difference in this robustness performance from the two algorithms, χ^2 (1, N=20)=13.3, p=0.0003. Even when the MCMC MRF algorithm did succeed, the average RMS error was much greater than the error for the MPS tracker; although this error could only be measured across the 2 available correct runs, and so should perhaps be interpreted with caution (the low significance of the accuracy difference reflects this caution, t(1)= 1.28, p=0.42) . What is certain is that the MCMC MRF algorithm irrecoverably lost at least one target on 8 of the 10 runs.

DISCUSSION

Loss of tracking during perturbations is caused by the sharp and sudden change of direction, and the effect is compounded by clutter causing the tracker to become distracted in the absence of a target measure. In the absence of clutter, traditional MCMC MRF tracking or even condensation would produce a satisfactory result on this sequence, by maintaining their original trajectories until the target rejoined its original path, and then recapturing this target.

Note that the RMS error for the MPS tracker is not much worse than that measured in the simple sequence of Experiment 1.

The success of the MPS algorithm for this sequence is likely to be because the global group motion keeps most of the particles largely on track during the perturbation phase, allowing them to rejoin the target when it returns to its original path.

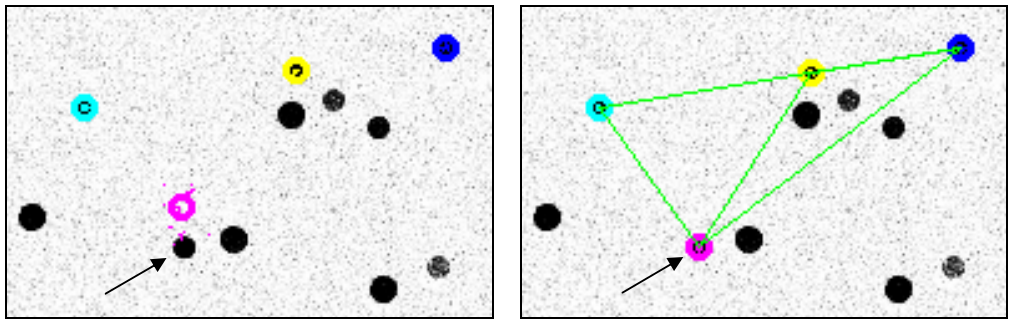


Figure 5.5 Khan et al.'s MCMC MRF algorithm (*left*) and the MPS algorithm (*right*). This frame is taken just as the perturbed target returns to its original trajectory. Note how the MPS estimation of position is correct, and how MCMC MRF has failed to recapture the target (the arrow is pointing to the actual target; the MCMC MRF prediction can be seen to be off the target in the left image). The green lines indicate the calculated correlations are greater than the correlation threshold, so the targets are grouped and considered able to share motion parameters.

5.5.4. Experiment 3: Occlusion

This sequence places one of the targets in the group in a simulated occlusion event. This in real life might represent a person or animal walking behind an occluding object. In the simulation, the third target is removed from the sequence

for 30 frames. This 'occlusion' period is marked by crosses in Figure 5.4. The results of this experiment are presented in Table 5.3.

	MCMC MRF	MPS
Correct runs	0	10
Average RMS error for correct runs (pixels)	-	0.767

Table 5.3. Table of tracking successes and RMS errors of successful runs (compared to the un-occluded ground truth) for the occlusion sequence

These results show that the MCMC MRF algorithm is very poor at maintaining tracking during occlusion where similar clutter is present. Conversely, the new MPS algorithm succeeded in maintaining tracking on every run, and with a low RMS error which was almost the same as in Experiment 2. There is a very high statistical significance in the robustness ('Correct runs') performances of the two algorithms (χ^2 (1, N=20) = 20, $p<0.00001$). RMS error was not calculated for the MCMC MRF algorithm as at least one target was irrecoverably lost on every run.

DISCUSSION

The MPS algorithm has good results because it keeps the estimation on track in the absence of reinforcing measurements from the image because the global motion parameters from the correlated targets prevent the peak in the probability distribution from spreading too far. This diffusion effect in standard particle filtering makes reacquisition of the occluded target somewhat hit and miss, and increases the likelihood the background clutter will be adopted instead. The effectiveness of MPS can be seen in Figure 5.6, where the arrowed target is occluded and its corresponding particles are not spread out as it is part of a group whose motion is very tightly coordinated.

As the dynamics of the occluded target are shared from the correlated targets, this mechanism will cope with the group turning corners as well as travelling in a straight line, as long as the relative velocities of the members remains the same.

This means the occluded target can still be tracked no matter what the dynamics of its motion sharing-colleagues, as long as their dynamics are representative of the motion exhibited by the occluded target. Even if the dynamics of the group *change* during the occlusion event, this theory still holds. Any algorithm which considers only individual motion will be sure to fail at such an occurrence.

Note again that the RMS error for the MPS tracker is similar to that measured in the simple sequence of Experiment 1, when no occlusion took place.

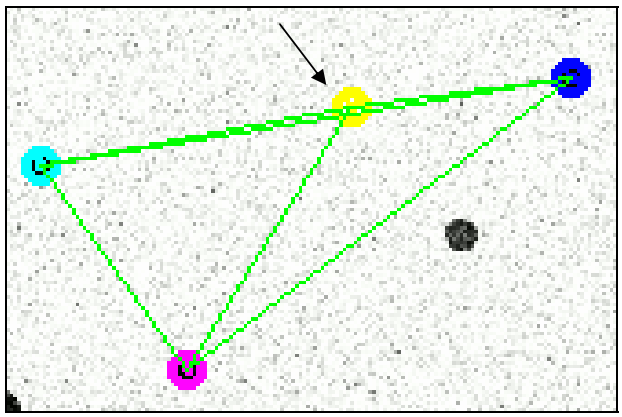


Figure 5.6. Frame taken from the output of the MPS tracker. The arrowed 'target' is in fact occluded at this time; the position is estimated based on the motion parameters shared from the correlated targets (these targets indicated by the lines).

5.5.5. Summary

From Experiment 1 it can be seen that the MPS algorithm is able to track the targets as accurately, if not more accurately, than its MCMC MRF counterpart. It performs very well during perturbation and occlusion events as has been shown in Experiments 2 and 3. Conversely, the previous best group tracking algorithm (MCMC MRF, as determined in Chapter 4) performed very poorly in such scenarios. These results suggest the MPS algorithm has the potential to cope with problematic scenarios that other algorithms that do not make use of social motion cannot cope with. A next logical test of the algorithm is to examine its performance under different measurement noise levels, as occlusion and certainly perturbation can be considered as special cases of such noise. This, therefore, will be the subject of the following group of experiments, followed in turn by some experiments on real world sequences.

5.6. Experiments with positional noise

It is the aim of these experiments to determine how well the MPS algorithm performs under varying positional measurement noise levels for all targets (experiment 1), and also where just one target in the group is suffering particularly heavy noise (experiment 2). The hypothesis is that when the targets' motions are correlated, the MPS algorithm's prediction will be less affected by noise. This is because the noise on the targets is independent and so noise on one target will tend to 'cancel out' noise on other targets. With one target under heavy noise, the situation is similar to the perturbation experiment in section 5.5.3.

Test sequences for this section were artificial, and were generated in a similar way to those used in the previous group of experiments (see section 5.5.1).

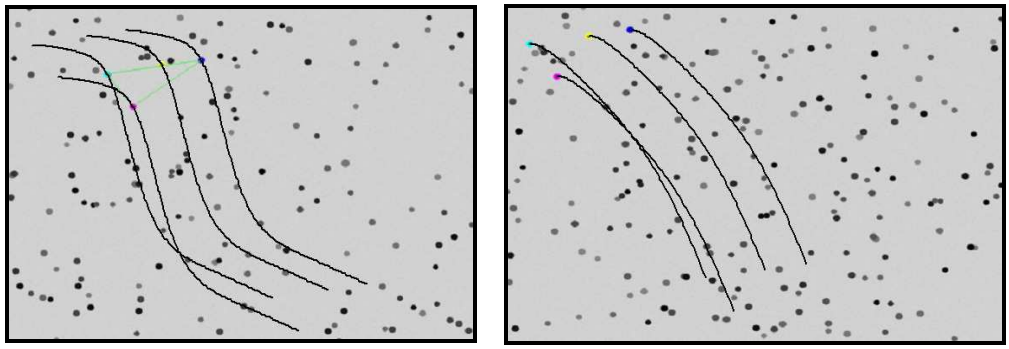


Figure 5.7 Paths and typical clutter used in the experiment for Sequence 1 (left) and Sequence 2 (right).

They again consist of a 'group' of target circles moving across a background of similar-looking clutter. Noise levels were applied to the motion of the circles in the form of normal (x,y)-displacement noise, to simulate positional measurement errors. Noise was applied independently to each target. The standard deviation of the noise was varied to test the algorithms under varying conditions. The background has Gaussian image noise (σ=4.0) added each frame. Two paths for the group were used. Sequence 1 consists of a smoothed ziz-zag across the image plane. The path for Sequence 2 is a curve. Sequence 1 consists of 265 frames, and Sequence 2 has 130. The paths that the targets follow are illustrated in Figure 5.7, along with some typical background clutter. These sequences simulate the

real-life scenario of bodies moving through a space in a gregarious manner, such as a group of friends moving through a crowd or animals foraging for food. 400 samples were used to represent the joint space for each tracker.

5.6.1. Experiment 1: All targets affected by the same level of noise

In this experiment all the targets are tested under the *same* levels of positional noise. This represents, for example, a noisy sensor taking the measurements.
The results of this experiment are presented below.

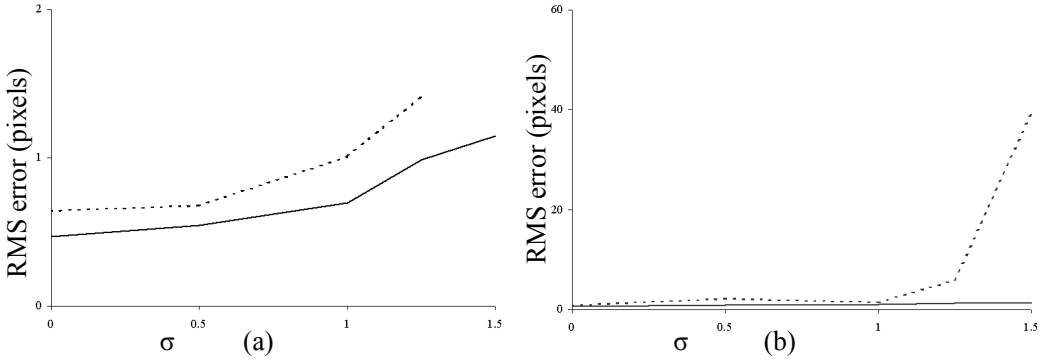


Figure 5.8 RMS errors between the MPS (solid) and MCMC MRF (dashed) tracking algorithm results and the groundtruth data for Sequence 1 (a) and Sequence 2 (b). Noise levels for the group increase along the x-axis. The MCMC MRF algorithm in Sequence 1 could not produce meaningful RMS results for displacement noise of σ=1.5, as tracking of the target was completely lost. Once tracking is lost, RMS errors can lose their meaning, hence why this data was not plotted in graph (a).

DISCUSSION

Figure 5.8 shows that at lower noise levels, the two algorithms are comparable. As noise levels increase, the errors on the MPS algorithm remain low, as the targets' motions are kept on track by using information from fellow targets in the group. This result demonstrates the effectiveness of this algorithm in the presence of measurement noise.

5.6.2. Experiment 2: One target affected by more noise than the others

It was shown in the previous experiment that when all targets are affected by noise, the MPS algorithm is robust to this noise as the motion of the group as a whole is used to partially cancel out some of the noise. In this experiment, the effect of the algorithm will be tested where one target is affected by more

positional noise than the other targets in the group, across varying base noise levels for the group. This is to simulate the real world situation where one target in a group is forced to follow a more erratic path than the others, or where one target looks more similar to the local background clutter. The experiment has three stages. At each stage, a different, fixed level of noise will be applied to three of the four targets, and an increasing amount of noise will be applied to the fourth target – called the 'test target'. At each step, the performance of the MPS algorithm will be compared to the MCMC MRF algorithm. The groundtruth used represents the case where no noise is present; therefore, an ideal tracking algorithm with zero RMS error would effectively be filtering out all the noise.

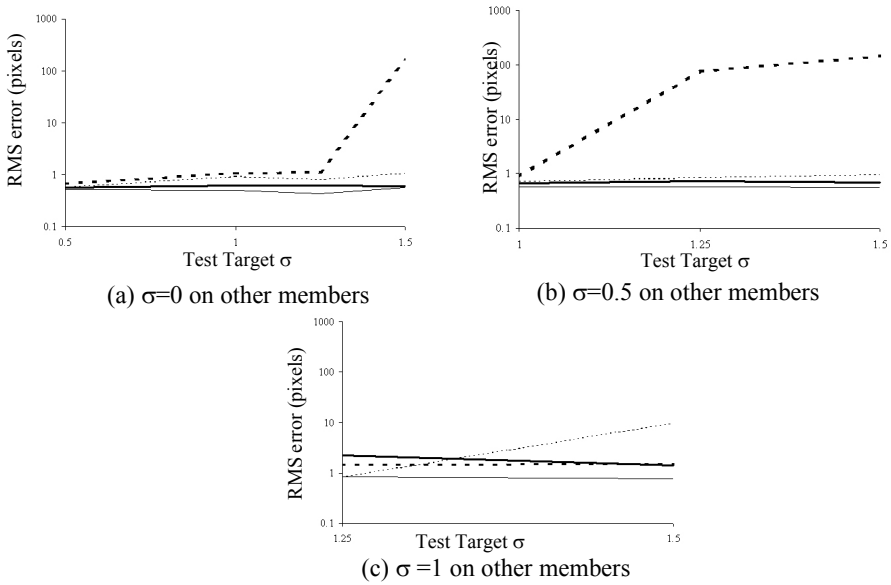


Figure 5.9: Sequence 1 RMS error results. The three graphs correspond to 3 different noise levels on the non-Test targets.

Thick (MCMC MRF): *dotted* = Test Target, *solid* = Median of other 3 targets
Thin (MPS): *dotted* = Test Target, *solid* = Median of other 3 targets

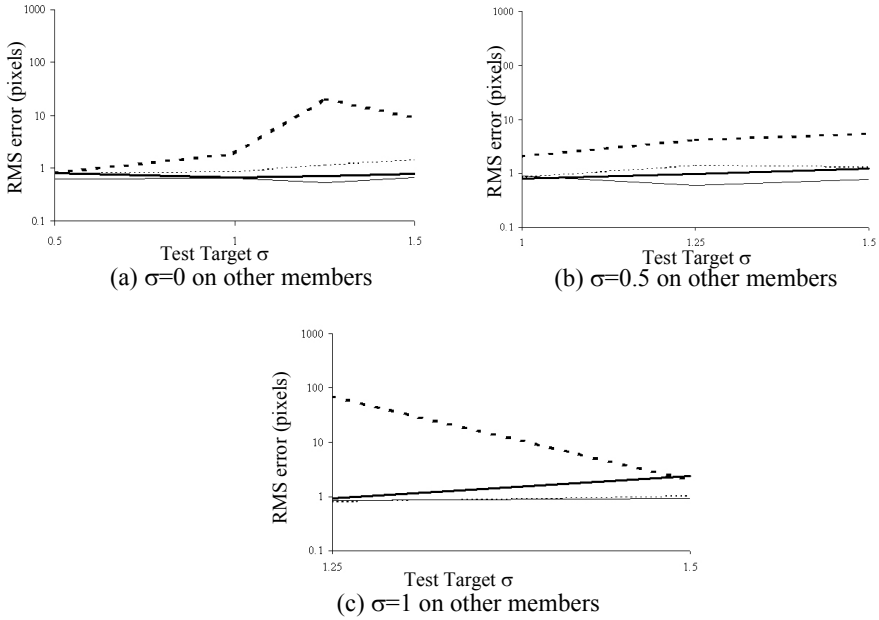


Figure 5.10: Sequence 2 RMS error results. The three graphs correspond to 3 different noise levels on the non-Test targets.

Thick (MCMC MRF): *dotted* = Test Target, *solid* = Median of other 3 targets
Thin (MPS): *dotted* = Test Target, *solid* = Median of other 3 targets

It can be seen from Figure 5.9 and Figure 5.10 that tracking is particularly poor for the test target when not using MPS. This is to be expected, as this is the noisiest target. Conversely, the test target under MPS tracking is generally (except in Figure 5.9(c)) located with only a small increase in error over the other 3 targets in the group. In fact the high MPS Test Target error in Figure 5.9(c) for σ =1.5 was caused by the tracker failing to catch the target at the beginning of the sequence, before MPS is functioning fully (while the correlation window is still being built). Therefore this error could not have been corrected by MPS.

The tracking performance of the remaining 3 targets in the group across the sequences shows that the MPS algorithm tracks more accurately than without MPS even with one target with extra noise. This is to be expected following from the results presented in Figure 5.8.

5.7. Experiments with real sequences of social animals

It is important to show how the algorithm performs with collections of real life, social targets. Therefore, the algorithm was tested on a number of sequences involving ducks, including three sequences used in the experiments of the previous chapter for comparison. The sequence numbers correspond to the sequence numbers in Chapter 4.

5.7.1. A simple sequence, sequence 3

This sequence was tested with the MCMC MRF algorithm in section 4.10.3 in the previous chapter, which was found to successfully maintain track of all the ducks in the sequence with low RMS errors. For comparison sake, the same sequence will be tested with the MPS algorithm with the same initialisation parameters.

Once again, the tracking was found to be robust and accurate, as can be seen in the residual graphs in Figure 5.11:

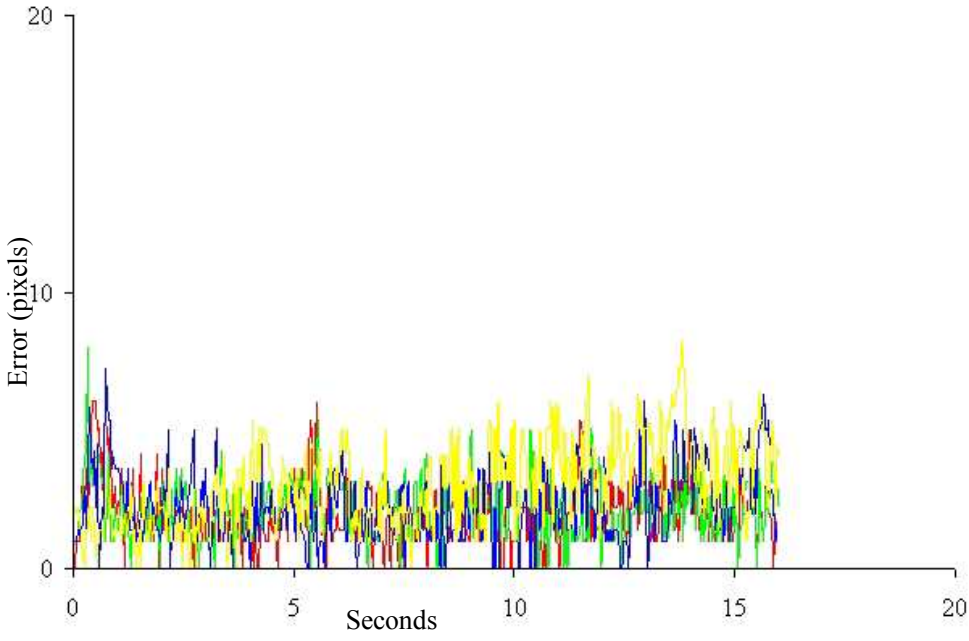


Figure 5.11 Residuals between groundtruth and actual data for Sequence 3 for the MPS algorithm.

As all targets were tracked successfully, comparing RMS errors between MPS and MCMC MRF becomes a meaningful test. The errors for the two algorithms are presented below:

Algorithm	Red RMS (pixels)	Green RMS (pixels)	Blue RMS (pixels)	Yellow RMS (pixels)
MCMC MRF	2.5	2.5	2.7	3.3
MPS	2.4	2.4	2.6	3.4

Table 5.4 RMS errors for the two algorithms for this sequence

These results show that the MPS tracker is marginally more accurate on three out of the four runs, although it cannot be said that it is much of an increase ($t(6)=0.16$, $p=0.87$). However, it is important to note that it performs *no worse* than the comparison algorithm, an important property of any new tracking methodology. This is consistent with the experiments using artificial data (see sections 5.5.2 and 5.6.1) which suggested that the MPS algorithm was slightly more accurate than the MCMC MRF algorithm when tracking coordinated groups of targets.

It is interesting to note here that by correlating motion based on *speed*, oncoming targets can be considered to be part of the same group (see Figure 5.12). This can be both an advantage and a disadvantage. If the targets join up and move together when they become close to each other, then sharing the particles' motion parameters between all the ducks should aid the tracking as their motion becomes more similar. If, however, they pass by one another then this may hinder tracking as some of the particles will want to travel in the direction of the members which are moving in the opposite direction. However, this latter case is the situation in this test, but the results in Table 5.4 show that it has no real impact on the tracking accuracy. In fact, MPS is generally more accurate then MCMC MRF, although as seen in Table 5.4, not by a great amount. This slight increase in accuracy may be from the particles whose motion is derived from the targets which *are* moving in the same direction. It can be seen that the yellow tracker performs the worst for MPS in Table 5.4; this may be because it is not always a member of a group (see

Figure 5.13 for example) and so does not always have a coordinated fellow target to share motion information with.

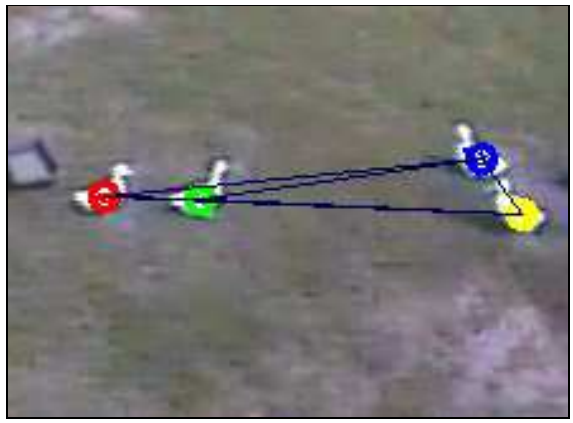


Figure 5.12 Example output frame from the MPS tracker. Although the ducks are moving in opposite directions, their speeds are correlated and so are considered part of the same 'motion group'.

Figure 5.13 Example output frame from the MPS tracker. Three of the four targets' movements are correlated above the threshold level, and hence three of the targets are grouped together as indicated by the lines – this is an example of automatic grouping as a consequence of the algorithm.

As can be seen in Figure 5.13, one consequence of implicitly detecting when animals are moving in a coordinated fashion is the automatic division of the targets into groups based on how correlated their motion parameters have been. The fact that the yellow target in Figure 5.13 is not a part of the group is nothing to do with distance, and is solely based on how correlated the targets' movements have been over the sliding time window. This kind of effect allows groups to be automatically determined that are not obvious from the sequences (see section

5.4.2 for an example of how two ducks were found to be moving in a very similar way by using this correlation mechanism). Of course, the kind of group that is determined is a function solely of the metrics you use to define a 'group' – *this* grouping is based on correlated speed metrics and may deduce groups that are not the same as those that would be selected by a human, for example. The issue of *what defines a group* is an over-arching one and can only currently be answered by simplifying the definition to a set of measurable parameters.

In the MPS algorithm, motion information is only shared with targets in the same group, so the joint state of the tracker can represent several sub groups, each moving independently, and the algorithm will automatically decide which targets the motion parameters should be shared between.

5.7.2. A typical duck monitoring example, Sequence 1

The MPS algorithm was also tested on Sequence 1 from Chapter 4. This sequence was tested in the previous chapter with MCMC MRF in section 4.10.1, but for this experiment the effect of different sample sizes will be examined.

The sequence showed a group of ducks being herded through an outdoor environment; see Figure 5.14 for some example frames.

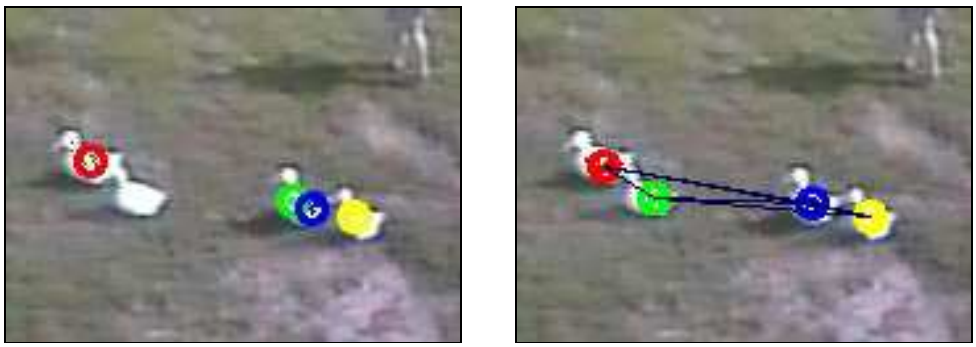


Figure 5.14. Example details of frame 180 from the duck sequence. Left, the MCMC MRF tracker fails to track correctly. Right, all four targets correctly located using the MPS algorithm. Lines indicate correlated motion.

The sequence is challenging because the ducks move in a complex way, moving close to each other and other ducks at times, making loss of tracking on clutter or similar targets a potential danger.

The RMS errors in Table 5.5 and Table 5.6 were calculated by comparing the tracking results against a ground truth every 10 frames for the sequence. This experiment was repeated 10 times for each of the MPS and MCMC MRF versions, and for both 600 and 500 samples representing the joint state space. The number of tracking estimations located off the correct target in the last frame was also recorded for each of the repetitions, and the totals presented in the last column in the tables above. A median of RMS values is used as the data is not normally distributed due to high tracking errors when a target is completely lost.

Algorithm	Target 1 RMS (pix)	Target 2 RMS (pix)	Target 3 RMS (pix)	Target 4 RMS (pix)	Misplaced tracking estimates
MPS	2.25	1.9	2.3	1.8	6
MCMC MRF	2.1	32.5	2.45	1.9	12

Table 5.5. 600 samples. Median (over 10 repetitions) RMS errors when compared to a ground truth, and the total number of trackers misplaced for all repetitions

Algorithm	Target 1 RMS (pix)	Target 2 RMS (pix)	Target 3 RMS (pix)	Target 4 RMS (pix)	Misplaced tracking estimates
MPS	2.2	1.95	2.45	1.9	4
MCMC MRF	2.1	33.15	2.6	1.95	11

Table 5.6: 500 samples. Median (over 10 repetitions) RMS errors when compared to a ground truth, and the total number of trackers misplaced for all repetitions

The results indicate that the addition of the MPS algorithm makes the tracking more robust (for misplaced tracking across the two algorithms, χ^2 (1, N=80)=2.58, p=0.1 for 600 samples, and χ^2 (1, N=80)=4.02, p=0.04 for 500 samples) and slightly more accurate as measured across the targets which did not lose tracking

completely, targets 1, 3, and 4 ($t(57)=0.30$, $p=0.77$ for 500 samples, $t(48)=0.22$, $p=0.83$ for 600 samples). This increase in accuracy, however, is not a statistically significant amount, but again it can be said the MPS algorithm is performing no worse than the MCMC MRF method.

Although the MCMC MRF algorithm does have a mechanism for handling similar target interactions, it can be seen from the results (Table 5.5 and Table 5.6, MCMC MRF rows) that this alone does not guarantee success, as the errors for target 2 are high.

It can be seen that the new MPS algorithm is in most cases more accurate, though by a small amount. The most significant result is that maintaining *target identity* (robustness) throughout the sequences is much better with MPS; in other words, the ability to say where a duck from the first frame ends up in the last frame is much improved. This type of 'robust' result is important for behavioural studies, where maintaining the identity of the target throughout a sequence is often more important that having a high positional accuracy, as targets will typically have to be tracked for long periods of time. In a group situation it can be easy for tracking location estimates to swap targets, and this is the main error that causes the loss of identity. MPS tracking helps avoid such errors by helping the tracker stay located over the correct target.

With 500 samples (Table 5.6) as opposed to 600 samples (Table 5.5) the RMS errors generally increase, as might be expected (although with low statistical significance: discounting target 2 again, for MPS across the sample levels $t(58)=0.35$, $p=0.7$ and for MCMC $t(52)=0.34$, $p=0.7$). Any increase in error may be because fewer samples mean a less accurate representation of the probability space. Note however that the MPS algorithm still manages to have fewer misplaced tracking estimates at the end of the sequences than the MCMC MRF algorithm. In this example, both algorithms misplace targets slightly less frequently with fewer samples. This is likely to be due to smaller numbers of particles meaning less of the state space is explored, making particles less likely to fall on and start tracking incorrect similar targets. Of course, decreasing the

number of particles can cause complete loss of tracking if too few are used. However, when in group situations with fewer particles, MPS sharing more efficiently represents the state space as the particle distribution is partially guided by social effects.

Interestingly the RMS accuracy for this experiment is better generally than for the MCMC MRF result in 4.10.1. This may be for a number of reasons. The ground truth used for this sequence was an early version where only every tenth frame's data was captured; perhaps the manual generation of this data was more accurate and had a lower base level of noise? This could contribute to the apparent increased accuracy here.

The average processing speed per frame for the MPS algorithm was 0.06s with 600 samples and 0.05s with 500 samples. With the MCMC MRF algorithm, the processing time was 0.05s with 600 samples, and again 0.05s with 500 samples.

5.7.3. A complex flocking situation, Sequence 2

During the algorithm tests in the previous chapter, Sequence 2 proved challenging to track (see 4.8.6 for Condensation's example results and 4.10.2 for the MCMC MRF example results). There are multiple reasons for this, including:

1. The ducks accelerate rapidly from being motionless to full speed
2. The ducks occlude one another at times
3. The animals move close to one another

Results from the previous chapter suggest that MCMC MRF and Condensation performed about as poorly as each other, although over many repetitions MCMC MRF could be expected to perform better due to its interaction-handling motion model.

This sequence represents a highly social behaviour: the ducks appear to be startled by something and exhibit a group flocking motion heading towards the right of the enclosure, presumably away from whatever startled them. It is predicted that the MPS algorithm, with its ability to make use of the motion of

other members of the startled flock, will outperform the MCMC algorithm. This test will not look at accuracy of tracking, but simply robustness. The previous chapter demonstrated how much of a challenging sequence this is to track, and so ending the sequence on the correct target will be considered as a success for the tracker. 12 targets will be tracked, and as the sequence was found to be so challenging in the previous chapter, the initial parameters will be tuned by hand: the velocity process noise is increased from 0.15 to 0.3 to help capture the rapid acceleration, and the targets that provided the MCMC MRF algorithm with the greatest challenge before – pink, black, yellow and blue targets – were manually given an initial velocity appropriate to their observed behaviour. The window over which correlations were calculated was decreased to 10 frames (~0.4 seconds) as the action is so fast paced. Also, the correlation threshold was raised to 0.9: as all the ducks are moving in roughly the same manner, a higher number was needed to differentiate those more highly coordinated in order to improve the automatic grouping capabilities.

The MPS and MCMC MRF algorithms were tested on this sequence with the same initialisation parameters with 300 samples per target (3600 therefore in total) – this produced very good tracking results in both cases. Previous experiments suggested that the MPS algorithm would outperform the MCMC MRF algorithm using smaller numbers of samples, therefore the total number was decreased to 3000. This made the tracker more unstable producing less than perfect results, and the comparison was therefore run at this level to simulate only just having enough samples to track the targets – i.e. to maximise speed. Both algorithms attempted to track the sequence 10 times, and the results are presented in Table 5.7 below.

	MPS	MCMC MRF
Number of runs where at least one target was tracked unsuccessfully	3	10
Time per frame (seconds)	0.2	0.2

Table 5.7 Comparison of the MPS and MCMC MRF algorithms for sequence 2

The results show that in this complex sequence the MPS algorithm far out-performed the MCMC MRF algorithm, with only 3 failures out of 10 runs compared to complete failure for the MCMC MRF runs ($\chi^2(1, N=20)=10.8$, $p=0.001$). Both algorithms ran at the same frame rate, which was quite low as so many samples were needed due to it being a complex sequence with 12 targets.

The correlation threshold value was found to be quite significant, as was the correlation window size, and both had to be tuned to the particular action taking place: a small time window for the fast action, and high correlation threshold to better split the targets into groups as they were moving in approximately the same way. As well as the number of samples, these parameters seem crucial to the success of the MPS algorithm and warrant further investigation in future work.

5.7.4. Alternative flocking example

A group of eight flocking ducks were tracked through a short but fast sequence of 30 frames as they made their way across a closed arena. This is a different kind of sequence (see Figure 5.15) from that used before (being from video captured during the pilot session of video capture), but no special effort was made with regard to determining a new measurement model or initial parameters in order to see how the algorithm coped with adaptation to new scenarios. Ducks were modelled as white circles, using the methods employed in the previous experiments. This makes an already complex sequence involving fast motion and some occlusion even more challenging for both trackers.

Figure 5.15 shows the final frames from the two algorithms. Note how the MPS-extension leaves the target estimations accurately placed in the final frame, as was the case throughout the sequence. There are clear errors in the output of the MCMC MRF algorithm. These are likely caused by local variations in the animals' velocities (introduced by their characteristic gait), which shake off the tracker.

Previous experience shows the performance of MCMC MRF can be expected to increase with the number of samples used, though Motion Parameter Sharing provides greater robustness at sample numbers for which MCMC MRF fails (in this sequence both trackers used 1000 samples and only the MPS tracking can be considered successful). The ability to track successfully using fewer samples means that tracking can be accomplished more quickly and/or more targets can be tracked with the same computational resource.

Figure 5.15 Final frame using the MCMC MRF algorithm (*top*) and the MPS algorithm (*bottom*). Note that the estimated positions (circles) are centered over the targets in the MPS version but with MCMC MRF there are clear errors (indicated by arrows).

5.8. Using the MPS algorithm to improve tracking of pig data

The kink features which were tracked in Chapter 3 in order to locate the P2 position on the back of a pig can be thought of as behaving as a group. Their motion has to be coordinated in some way as they are attached to each other via the pig's body. While this is not a rigid structure, it can only change its shape in a fixed, if large, number of ways. Some tracking methods can make use of the fact the object is semi-rigid (Tsutsumi and Kita 2002), but treating the features as a social group is advantageous because it pre-supposes no specific geometric shape on the animal: the kink points could be in any configuration and the MPS algorithm would still recognise them as moving as a group. It is hypothesised that the occlusion event caused by the robot arm obscuring one of the kink feature points should present less of a problem to the MPS algorithm than to the MCMC MRF algorithm. This is because motion information from the three features which remain visible can be used to guide tracking of the fourth, occluded feature. The practical advantage of this is that an estimate of P2 position might still be able to be calculated during occlusion, and after occlusion there is an increased likelihood that the tracker will begin tracking the correct feature again.

The test sequence itself replaced the kink points with circle targets at the positions the kink features appeared at, to enable the existing measurement model to be used to track the targets. In this respect, the sequences resembled those of the artificial sequences described earlier in this chapter, with the exception that the targets move according to motion information taken from a real sequence of a pig in a feeding stall. Gaussian image noise was added to the images as before, but no similar target clutter was used, as in the real sequence the kinks are relatively easy targets to detect on the boundary with no similar clutter on the background or foreground.

During this sequence, the green target was removed from the images between frames 100 and 200, simulating a four second window in which the robot arm is

activated and completely obscuring the kink point while taking a P2 ultrasound reading (refer back to Chapter 3 for details).

The following section presents residual error graphs for the four kink targets as measured throughout the 265 frame sequence. 10 repetitions of the tracking were performed for each of the MCMC MRF and MPS algorithms. When one of the target estimates leaves the image, tracking is cancelled as in real life this would mean a completely unpredictable prediction of the P2 point, which uses all four kink point locations in its model. This situation is represented by the bold vertical line at the end of the graph. 1000 samples were used to represent the statespace in both algorithms.

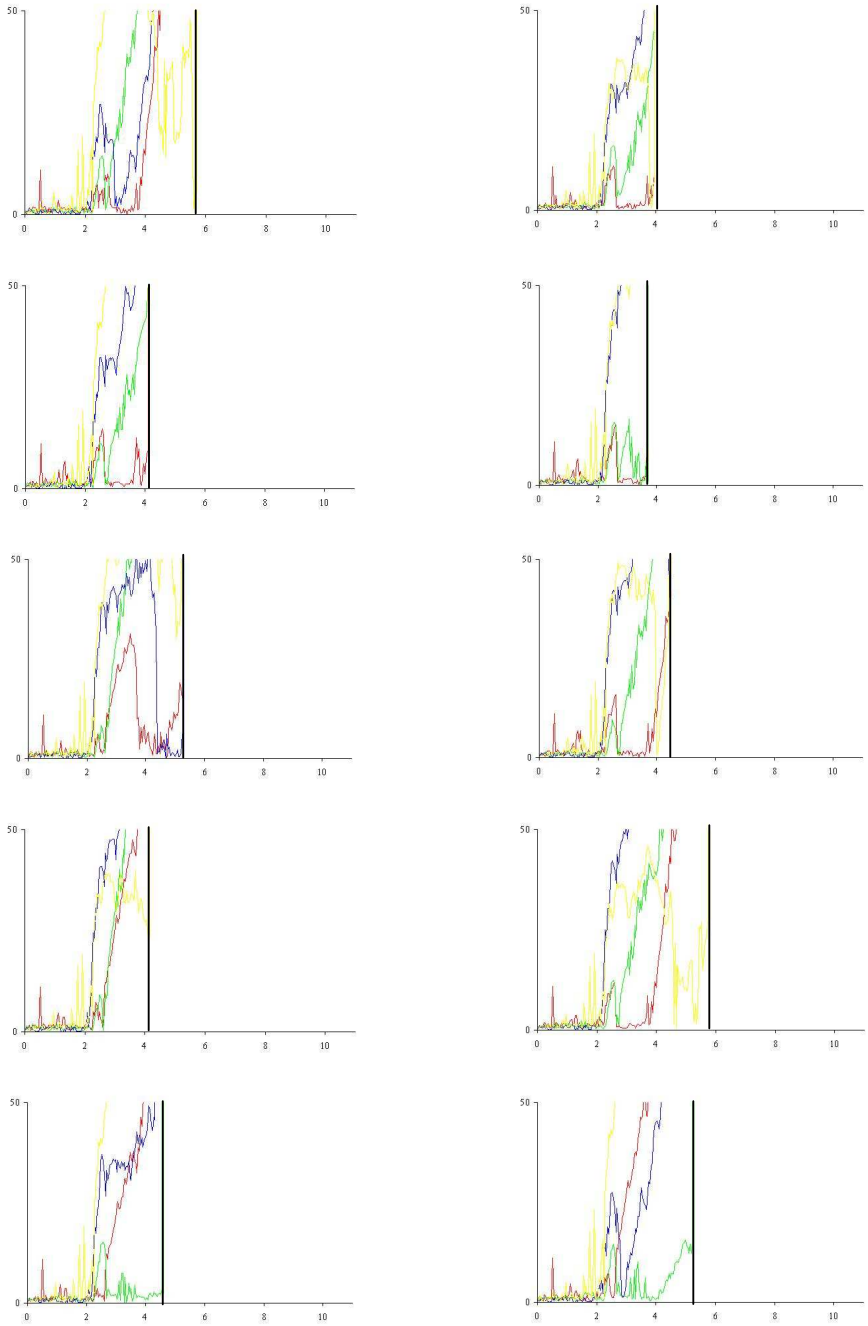


Figure 5.16 Graphs of residual errors in pixels (against time in seconds) for the MCMC MRF algorithm compared to the groundtruth when tracking data representing kink feature positions of a feeding pig. The graphs show 10 repetitions. The green target is occluded between 4 and 8 seconds, representing a robot activation event where the robot arm obscures a kink point. Vertical bars represent the points where tracking was cancelled due to an estimated kink position leaving the image, i.e. an unrecoverable error.

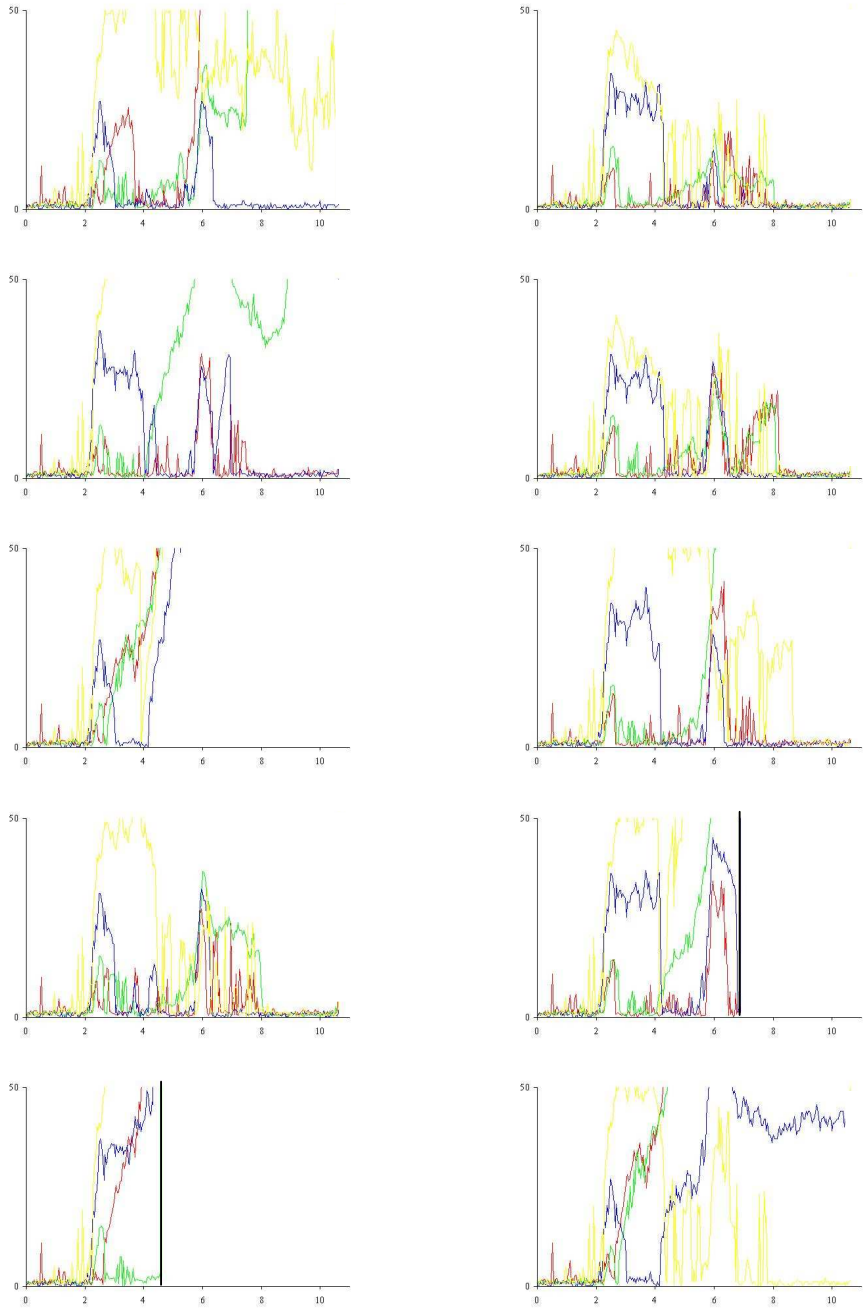


Figure 5.17 Graphs of residual errors in pixels (against time in seconds) for the MPS algorithm compared to the groundtruth when tracking data representing kink feature positions of a feeding pig. The graphs show 10 repetitions. The green target is occluded between 4 and 8 seconds, representing a robot activation event where the robot arm obscures a kink point. Vertical bars represent the points where tracking was cancelled due to an estimated kink position leaving the image, i.e. an unrecoverable error.

DISCUSSION

From these results, it can first be seen that the MCMC MRF algorithm rarely manages to track all four targets much beyond 4 seconds: this is just after when the occlusion simulation phase starts. The MPS algorithm typically manages to track the sequence for much longer, only having to cancel tracking on two of the ten trials (though for at least one more the final residual values were very high). For three of the trials, the MPS tracker has managed to keep all the location predictions on the targets at the end of the sequence. Even these results for the MPS algorithm, however, have quite large residual errors for one or more of the targets, typically during occlusion. This was caused by a number of things, including the tracker swapping the targets' identities, or simply wandering off-target but being held quite close to formation by using the correlated motion of the others as a guide. Another issue may be that the motion of the kink points is not being correctly predicted by the motion sharing algorithm, perhaps because the front end of the pig can move in ways unrelated to the back end, for example. This large error would propagate through to the P2 location estimate in practice, and so would be likely to cause errors for the sensor placement.

By 'socially constraining' the location of the occluded green target during occlusion, the MPS algorithm has the potential to maintain a prediction of the P2 location whilst the robot is activated, albeit with a relatively high positional error. This would allow an updated estimated of P2 position to be sent to the robot during the activation phase, effectively enabling online tracking: a scenario that was not possible with the methods used in Chapter 3. These results suggest the MPS algorithm may produce increased tracking performance over MCMC MRF, allowing a tracker the chance of recovery after occlusion by the robot, and possible continuous P2 prediction during the occlusion phase. However, the MPS tracking can still produce large RMS errors, mainly during the occlusion phase. This algorithm, although offering theoretical improvement in robustness over algorithms such as MCMC MRF, is likely to be too inaccurate (as indicated by the high errors) to enable practical implementation in the sensor placement system at this time. Further work would be required to ascertain whether the algorithm could be successfully implemented in this scenario.

5.9. Discussion of the performance of the MPS algorithm

This chapter has described a new algorithm which combines mixed state particle filters and a Markov chain Monte Carlo sampling mechanism which allows the motion model to use parameters from correlated targets which have been moving in a similar fashion. This was incorporated with a spatial interaction model to prevent target estimations coalescing on the best target measurements during interactions.

The MPS algorithm performs as well as, if not slightly better than the MCMC MRF algorithm when tracking along simple paths (section 5.5.2) and when all the targets undergo the same positional noise levels (section 5.6.1). Performance during a serious path perturbation (section 5.5.3) suggests the MPS algorithm out-performs the MCMC MRF algorithm with both a small increase in accuracy and a large improvement in robustness. The same can be said for sequences where an occlusion event occurs (section 5.5.4).

Section 5.6.2 demonstrated the positive effect the MPS algorithm can have on tracking quality with groups where one targets is experiencing more displacement noise than the other members of the group. This would be useful for tracking groups where one target moves in an erratic fashion, but still with the group, such as a lame animal in a group, or where one target more closely resembles the background clutter than the other targets, producing more erratic measurements. Additionally, the particular target suffering from the additional noise need not be fixed: the noise could affect all the targets in turn. Such a situation might occur with a group of pedestrians walking down a street, with a number of obstacles in each of their paths. The results suggest the MPS tracking could use the motion of the less noisy paths to guide the motion of the more noisy ones for any period of time.

The results from the experiments that use real-life sequences of ducks illustrate that the MPS algorithm can offer a slight improvement in accuracy over MCMC MRF tracking, although this is not very significant statistically. At worst, the

MPS algorithm can be considered comparable to the MCMC MRF algorithm in terms of accuracy. Where the MPS algorithm does stand out is robustness, managing to maintain the identity of the targets throughout the sequence much more frequently than the MCMC MRF algorithm (e.g. Table 5.5, Table 5.6, and Table 5.7), often with high statistical significance.

One interesting side-effect that can be noted with these real life sequences is the ability of the algorithm to provide automatic grouping of similarly-moving animals. This is very obvious from viewing the videos, but can also be seen in Figure 5.13, where the drinking yellow duck is motionless and so not grouped with the other three, which are moving. Such grouping is represented by the lines in the images, which form graph-like structures of the groups. This grouping happens automatically and at no extra cost, as a natural consequence of the MPS algorithm. Of course, collections of objects can be 'grouped' in many different ways, and this particular method uses a measure of how well the animals' speeds have been correlated to effectively assign group membership. This could easily be changed if the algorithm were to correlate over any other parameter, and a distance metric could be incorporated if the more traditional method of grouping neighbours within a certain distance was required.

The final real-life duck sequence demonstrates the potential generalisability of the algorithm to other domains with different kinds of motion and camera angles. The pig-kink work suggests the MPS algorithm does offer potential in other less obvious domains, though the algorithm still struggled with the tracking, producing high errors at times.

Although all the results are generally promising, the algorithm is not without its limitations. During occlusion, for example, a tracker's motion model for a target can only be as good as the other targets' motions in the group, which may at any point be no longer representative of the occluded target's motion. If the occluded target begins moving in a different way to the other targets, then the MPS algorithm is stuck. This is why the algorithm needs sequences featuring social action in order for the best advantage to be made of the motion of the other

members of the group. Generally, if the motion of the occluded target becomes uncorrelated during occlusion, then the MPS algorithm will lose track; but then so will the MCMC MRF algorithm *and any other tracking methodology* that relies on direct observation of the targets.

One situation in which the MCMC MRF algorithm may track more successfully during occlusion is when the all the targets have been moving as a group beforehand, but then during occlusion the occluded target moves straight through the occlusion, maintaining its original trajectory, whilst the other targets move in a different direction. This is illustrated in Figure 5.18:

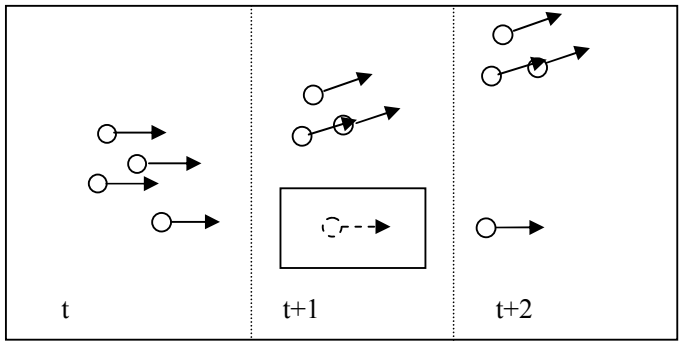


Figure 5.18 Diagram to illustrate a case where MCMC MRF would succeed and MPS would fail. As the motion of the group and the occluded target become different during occlusion (at t+1), MCMC MRF's best estimate of last known motion would be a better strategy than MPS's sharing of the group motion parameters.

This problematic case, however, can be remedied by tuning the split between the number of particles which share motion parameters from another target, and the number that use their own motion. Essentially, this is done by changing the probability that a correlated motion model is accepted for each particle: if it is not, then the particles' own internal motion is used. Particles that use their own internal motion parameters move in essentially the same way as those in the MCMC MRF algorithm. This would produce two clouds of particles: one following the internal motion of the occluded target, and one following the motion of the previously-correlated group. Tracking should be resumed when one of the clouds receives a target-quality measurement again. So the MPS algorithm can

overcome the stated problem this way, although future work is needed to determine the optimal probabilities involved, and this may in turn depend on the degree of gregariousness of the targets in the sequence. Determining the true level of gregariousness is clearly a key issue, and indeed this may be a very subtle effect which is not easily quantised.

One downside of this approach, however, is that having a more dispersed particle set produces more chances of one of the samples fixating on target-similar background clutter. One advantage of the MPS algorithm is that the particles' motion was more constrained and so the effect seen in Figure 5.2 was lessened. Allowing more particle clouds to follow different motion models for longer periods of time begins to enter this over-diffusion territory again, but at least each of the clouds is guided by a feasible motion model this time, rather than random motion with drift.

If there is one criticism of the MPS algorithm it might be that it potentially has an averaging effect on the motion of the individuals within the group. Consider a group of five targets, all exhibiting correlated motion and hence considered a group. If large enough, a number of samples in the sample set will produce estimated locations using motions for each of the targets from each of the correlated targets. The exact number of such 'socially derived' particles depends on how well the relevant targets are correlated. If each target is correlated with every other target r=0.6, then from equation (15) there is a ((1/4 x 0.6) x 4) = 0.6 chance that as each sample is propagated forward, the target processed forward will use a correlated fellow-target's motion parameters. Therefore, it can be expected that 60% of the final sample set will be generated using correlated knowledge, and 40% using internal motion information. Such a collection of particles represents all likely movements of any target, taking into account the motion of all targets it has been correlated with in proportion to the strength of this correlation. Therefore, this is not so much an *averaging* function as a way of shaping the exploration of the state space guided by social knowledge heuristics. An averaging method would shape the statespace exploration only in the direction of the average group motion, rather than taking into account the movements of all

correlated targets. It is this powerful, guided exploration that explains the success of the algorithm with fewer sample numbers than are required with the MCMC MRF algorithm.

An experimental issue was raised when deciding on what to consider a successful tracking of a perturbation. It was considered in Section 5.5.3 that the perturbation should be considered as noise on the path, and so the tracker should not follow the target *through* the event, but should recapture the target *after* the event has finished. In some situations, this perturbation may be a known and required feature that is needed to be tracked. However, in such a situation, the motion of the group cannot be considered as social, and so social motion algorithms like MPS should either not be used or a more appropriate correlation parameter should be chosen.

One implementation issue is with the way the correlations are calculated. As a sliding window is used, and within this an iteratively updated mean is calculated, there is an error accumulation in this mean calculation. This may in turn affect the accuracy of the correlation calculation over time. Future work is required to determine the size of this effect. Using speed as a parameter over which to correlate produced good results in the experiments. However, it is not without its problems. Firstly, as no directional information is provided, targets moving towards or away from each other will be considered correlated in the same way if there speeds vary by the same amount. However, this is not necessarily bad depending on the domain; if targets tend to come together and then move together or repel each other, then there may be a benefit to them sharing each others motion parameters. This was illustrated in section 5.7.1.

One effect of correlating speeds, or any parameter for that matter, is that the parameter must vary over the calculation window in order for a reliable correlation value to be determined, as illustrated in Figure 5.19.

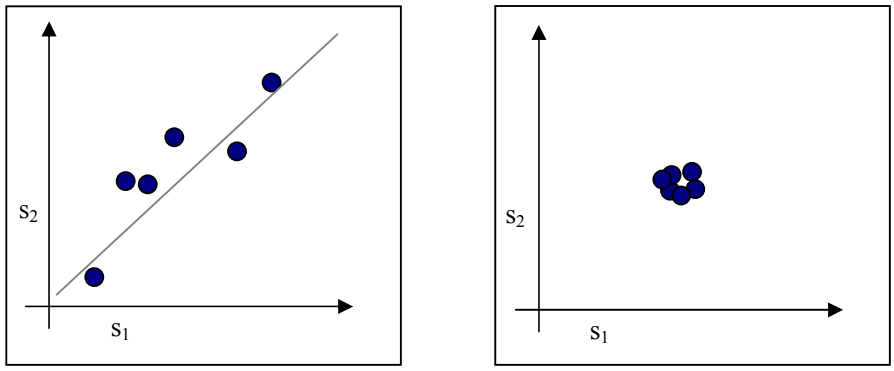


Figure 5.19 Example graphs of hypothetical speeds for target 1 (s_1) and target 2 (s_2). In the *left* graph, the targets are correlated and speeds have varied over time. Correlation is a measure of error from the best fit line. On the *right*, the targets are again correlated, but the targets have moved with an approximately constant speed. Here the correlation measure is unstable as the line of best fit can change rapidly, as the targets are all centred around a point.

The correlation coefficient is more unstable on the right image in Figure 5.19 as the line of best fit is likely to change drastically over time. However, this is not as large a problem as it first seems, as the correlation coefficient should still be large even if the line of best fit does change, as each data point should still be close to this line wherever it is drawn through these points. This, though, does beg the question of whether correlation is the correct statistical measure to use in this work. One alternative might be Mutual Information, which can measure the general dependencies between two variables and so may offer a more general solution. However, future work is required to assess the suitability of such alternatives, and for this work correlation, despite its limitations, was found to produce a metric which allowed a meaningful estimation of coordinated movement.

The speed of execution of each iteration of the MPS algorithm is comparable with the MCMC MRF algorithm, for the same number of samples (e.g. both algorithms run at approximately 0.05 seconds per frame for a 600 and 500 sample test in section 5.7.2, and at 0.2 seconds per frame with 3000 samples in section 5.7.3). Given that the MPS algorithm seems to perform better than the MCMC MRF algorithm with limited numbers of samples when social motion is present (as seen in sections 5.7.3 and 5.7.4), this suggests that the new algorithm may be able to run faster than MCMC MRF by using fewer samples to represent the state space.

However, as when no correlated social motions are present the MPS algorithm defaults to MCMC MRF tracking, the number of samples present should be equal to the number required for MCMC MRF tracking.

So, it has been seen that the MPS algorithm can greatly improve how robust tracking is for a comparable level of accuracy compared to MCMC MRF tracking, using social motion information to guide tracking.

Chapter 6: General Discussion and Future Work

6.1. Main Contributions

This thesis has produced two distinct and novel contributions:

- The combination of existing vision techniques to produce a novel system which is able to direct a robotic arm to the P2 point over a pig's back as it feeds.
- The development of a novel tracking algorithm, the Motion Parameter Sharing algorithm, which builds on the MCMC MRF and mixed-state particle filters to make use of the motion information present in a collection of targets exhibiting coordinated motion.

While distinct in their own right, these two components have both been empirically tested in the domain of animal monitoring. The second contribution has potential to feed back into the first, as seen in section 5.8. The MPS algorithm, although applied to animals, can be essentially applied in any domain where groups of targets need tracking, and if coordinated motion is present it will use this information to help guide the tracker. It also has the implicit ability to automatically group the targets which have been moving in a similar fashion.

The main achievements, limitations and possible extensions of this work are discussed in this chapter.

6.2. Achievements

The first part of the thesis produced the basis for a single animal monitoring system, in the process testing an implementation of a pig backfat sensing robot, tested in a real-world environment with live animals. This provides a proof of concept for using image analysis, robotics and an ultrasound sensor to measure pig backfat levels. From this work, it can be concluded that the robot can produce human-equivalent accuracy or better on 40% of sensor placements. This would allow at least 4 and potentially 40 placements of at least human accuracy per day, taking into account the number of times the pigs visit the feeder. For automatic monitoring, the sensor placement result suggests a sensing system to automatically measure pig backfat thickness is a viable alternative to the manual method in use at present.

The thesis then examined the tracking of multiple, similar animals. An examination of two major existing tracking algorithms for tracking multiple similar targets, and a description of the kinds of problems inherent in such work were then presented, concluding that the MCMC MRF algorithm outperformed Condensation, particularly in terms of robustness at maintaining individual identities throughout a sequence. For tracking multiple individual animals, then, it can be concluded that using particle filtering tracking with a joint state space and awareness of interactions produces a level of success suitable for tracking multiple individuals, though it still did not produce 100% robust or accurate tracking.

It was hypothesised that using social knowledge about how targets are moving might be able to increase the robustness of tracking further. The idea of using social motion to guide tracking was inspired by observing how the ducks flocked as a group when startled, and moved around the arena together in pairs on occasion in a highly correlated manner. Developing a tracking methodology to share social motion information between coordinated targets using the MPS algorithm was found to greatly increase the robustness of the tracking, especially in situations where one target in the group is completely occluded or follows a

perturbed path. It can also increase the accuracy of the tracking when the targets are exhibiting correlated motion and noise is present on the paths, though this increase is small. In summary, this suggests that in any situation where there is a social element to the motion, this extra knowledge can be exploited and lead to improved tracking robustness and similar or slightly better accuracy than the MCMC MRF algorithm. This suggests the MPS algorithm should generally improve the quality of tracking of multiple social targets.

The multiple tracking results suggests the MPS algorithm could be used for tracking multiple similar targets and be able to maintain their identities over longer periods of time than the MCMC MRF algorithm, which is an important result if it were to be used as the basis for an automatic monitoring system. Such a system would likely be required to track animals for long sessions, and perform analysis such as looking at outlier motion patterns: for this, individual identities must be maintained for as long as possible.

Together, the individual and multiple animal tracking systems provide a foundation for the potential development of automatic monitoring technology.

6.3. Future Work

6.3.1. Automatic backfat monitoring system

There are three main areas for development for the sensor placement system: two of which refer to locating the P2 point and one to the sensor deployment mechanism. When locating the P2 position, it was shown in section 5.8 that there is some potential to improve the system by being able to estimate P2 location when the robot is occluding one of the kink features – being able to do this would allow the robot to actively track to the current estimate P2 location after it had been activated. It was also shown that neither tracking algorithm tested in the multiple target tracking work sections of this thesis was really up to the job. More work would be needed to design a suitable tracking algorithm to make use of the information in all available kink locations that can predict missing data from an obscured kink point. The MPS algorithm showed potential here, but there are issues as to whether the kink points *are* acting as a coordinated group or not, and potentially issues of what parameters to correlate to allow them to be grouped accurately. This was beyond the scope of the illustrative example in section 5.8.

The second body of work required for predicting the P2 point is how to determine what the actual location is from the collection of P2 estimations collected per animal per day. As was seen in Chapter 3, the P2 point is a local minimum of fat thickness and so using the P2 location that produces the minimum fat thickness would be a good starting point (as long as placement errors were not too extreme). However, to test the validity of this, fat thickness readings would be required at each estimated point, which was again beyond scope of this work. This does however lead on to the third clear choice for future work: how to actually deploy a sensor and take a backfat reading. The sensor itself could be pushed vertically down with collision-forgiving pneumatics onto the pigs back, and mounted on a ball joint to allow the best alignment allowing for the contours on the surface of the pig. As was seen in Chapter 3, ultrasonic sensing systems exist which could provide a numerical reading using this method, and so one of these commercial

sensors would provide a suitable starting point, and readings could be manually corroborated by an operator standing by with equivalent kit.

6.3.2. Improvements to the current MPS social tracking scheme

PARAMETERS AND SCALE OF CORRELATIONS

The aim of the MPS algorithm is to allow targets that have been moving in some coordinated way over a period up until timestep *t* to then potentially 'share' from each others motion parameters during the processing of dynamics at time *t*. This is, however, a general framework and though specific decisions were made on what to correlate and over what time to correlate to enable experiments to be run during this work, much future work could be involved in determining the benefits and drawbacks of computing correlations across different parameters and timescales. What this means in real terms is deciding what defines a group of targets as 'moving together in a coordinated way', and how long should they move in such a way for before they can be considered to be moving together.

Correlating different parameters means that different types of motion will be used to assess whether the targets are moving as a coordinated group. In this work speed was used; therefore targets moving at a similar speed are considered to be moving in a similar way. Using velocity would produce a similar grouping to speed, except that the targets would have to be moving in the same direction as well. Other parameters which could be correlated include curvature (targets turning at the same rate are grouped), distance (targets that maintain the same distance from each other are grouped) etc., or combinations of these.

The size of the sliding window over which to calculate the correlation matrix is dependant on the time scale of the actions taking place. If something very slow is being tracked, the time window will likely have to be larger than if the targets are moving very fast. Also, the types of behaviour that can be captured vary with time scale. Targets might be correlated based on the current action they are executing, e.g. turning a corner, or over their entire route over the past number of minutes. This is also the difference between correlating two targets over their last action, e.g. a jump, or their last sequence of actions, e.g. a dance move.

Other mathematical alternatives to calculating correlations exist. One such alternative is Mutual Information, which can measure general dependencies as opposed to correlation's linear dependencies between two variables. Future work is required to determine the effects of using different methods to calculate the level of similarity on the targets' motion.

The actual implementation of the algorithm could be improved in a number of ways:

1. Optimization. The program, as implemented, runs in real time up to a point, depending on the number of samples used. Tracking many targets often leads the processing time to inflate, perhaps dropping to the equivalent of about 2 frames per second in serious cases. However, the algorithms were implemented with testing and accessibility in mind, rather than optimization. Implementing more efficient methods and data structures would be certain to drop the processing time down to allow real-time tracking of sizeable groups of targets with the MPS algorithm. To improve the efficiency of further algorithm development, future work should include the analysis of the complexity of the algorithm.
2. Measurement model. Using a circle for the geometry of the measurement model, although found to be adequate, did sometimes cause problems where two circles were allowed to fit on one target, thus producing two measurements from a single target. Designing a more specific model of shape might allow fewer high measurement responses to be inferred per target, but is also more orientation dependant. The problem of handling multiple measurements from single targets (and single measurements of multiple targets) is currently the subject of research in the multi-target literature (Khan et al. 2005b).
3. Dynamics. The dynamics of the animals modelled in the algorithms was based on observations and measurements from image sequences. There are much higher-quality ways of determining the model of target dynamics. One such method is to build a rough tracker to follow the motions in a simple training set, then using the learned dynamics of this

tracker to build a more competent tracker either to re-track the original training set more accurately or to track a more challenging training set. This cycle is repeated until suitably general motions have been learnt (Blake and Isard 1998).

Further Experimentation

Future experimentation should demonstrate the ability of the algorithm to maintain the tracking of an occluded target where the group dynamics change during this occlusion event. The effect of the MPS algorithm on non-social sequences should perhaps be examined more closely; although intuitively, if there are no correlations detected then the algorithm defaults to the MCMC MRF algorithm, so tracking quality of these algorithms should be equivalent in this situation. Similarly, the effect of non-coordinated targets becoming coordinated as the sequence progresses needs to be examined. Again, instinctively, as the levels of correlation increase between the targets, the algorithm should share motion between them more regularly. A demonstration should confirm that the algorithm can seamlessly handle such changes in how well coordinated the targets' motions are. The reverse effect should also be tested, where coordinated targets become uncoordinated: this may be a more challenging situation as the tracking has to 'break out' of sharing motion parameters. Work in this thesis has indicated that it should not present a problem, however. (e.g. in Figure 5.13 the yellow target leaves the coordinated group, and is still tracked successfully, albeit with a very small amount of increased error). The length of time the 'ghosting' effect occurs for during occlusions needs to be examined. In other words, this would look at how long an occluded target's estimation will be propagated for in the absence of a reinforcing measurement. Intuitively, the answer is forever: using only motion from coordinated targets will mean the level of correlation with those targets can only increase. Once these smaller experiments have been conducted, attention must be turned to the capability of the algorithm to track longer sequences as might be required by monitoring applications. However, this is only sensible to test once the targets in the shorter sequences can be tracked very reliably, and success in these shorter sequences of challenging scenarios

suggests the tracking will be successful over longer periods anyway, where most of the tracking is non-demanding.

Further experimentation should evidence the use the tracker makes of social motion. It should be possible to quantify how many predictions are made using social motion and how many are made using internal motion for various events in the video sequences. This would allow the effects of variations of the algorithm to be seen more clearly. It is perhaps worth considering new visualisation options for presenting this kind of data as well, so that these values are immediately accessible when viewing the output video.

It would be interesting to test the algorithm with flocking models, such as the Boids model (Reynolds 1987). Such models, if powerful enough, would allow the dynamics of the group to be carefully set and the effect of the algorithm carefully tested. However, this is limited entirely by the power and the accuracy of the models, which may not actually exhibit the behaviour one expects them too, especially as any true behavioural flocking rules are still out of our grasp.

Finally, it would be interesting to see if targets could be tracked accurately enough to be able to calculate the ground plane automatically, as previously work has suggested this to be possible (Bose and Grimson 2003).

A Note on Statistical Tests

It is recognised that chi-square tests have been used throughout Chapter 5 where the observed raw frequencies are often low. There is a school of thought which suggests that observed raw frequencies must be 5 or more, and if this assumption is not met Yates' correction must be applied. This produces a more conservative estimate of statistical significance. However, in this thesis all the chi-square results which would require this correction are all highly significant, so much so that the highest p-value after correcting the necessary tests is present in Section 5.7.3, where χ^2 falls from 10.8 to 7.9, raising the p-value from p=0.001 to p=0.005. This is still highly significant and demonstrates the robustness effects

are so strong that Yates' correction is not really required as the chi-square statistical tests are so significant anyway.

6.3.3. Extending the MPS social tracking scheme

EXTENSION TO LARGER AND VARYING NUMBERS OF ANIMALS

The group tracking work described here is designed to track the size of group as was used in the experiment, i.e. anywhere up to about 15 or 20 animals. Conceivably this work, given suitable processing power, could be used to track any number of targets as long as they have enough 'on image definition' i.e. can given suitable means be differentiated from background clutter, and are adequately separated from their co-targets. The MPS algorithm itself, however, is quite capable of running with any number of targets, given enough processing time as of course the number of samples must be increased with the number of targets.

Extending the algorithm to *varying* numbers of animals would allow animals to come and go in the scene. Although this did not happen with the video captured for this work, it could conceivably happen in other scenarios where the animals move in and out of a shelter, or in and out of the field of view of a camera in a multiple camera setup. Recent work (Khan et al. 2005a) has extended the MCMC MRF algorithm to be able to cope with varying numbers of targets, using Reversible Jump Markov chain Monte Carlo sampling which permits variable dimensional state spaces.

EXTENDING TO RECOGNISING GROUP MOTION EFFECTS

Once large numbers of targets can be reliably tracked over suitably long periods of time, then further processing can be used to perform some labelling or behaviour inference techniques in a variety of situation. There are two main flavours to the kinds of classifying that can be done. First, there is classifying the behaviour of a group as a whole, and second, classifying particular behaviours within a group. Group-scale behaviours which might be able to be recognised include suspicious groups, rioting groups etc., or in the case of ducks, the

categorisation of events into the kinds of categories listed on page 111. Clearly, the exact types of group behaviour that could be recognised depend on the domain and need further work to categorise these behaviours in a meaningful way. Recognising intra-group behaviours provide some more interesting scenarios. Targets moving in a different way to the majority of other targets being tracked might indicate someone acting suspiciously at a train station, a lame animal, or people stopping to look at something interesting in a shopping centre (perhaps a catchy window display). Being able to spot *group motion* effects would open the door to more interesting behaviour labelling – a group of people might be rushing towards one other person because they have fallen down, for example.

Recognising group behaviours is not restricted to groups of people or animals. There are plenty of examples where the targets may exhibit correlated motion but their domain's may not be obvious choices for application of the algorithm. What causes multiple targets to move in a correlated way could be some social aspect arising from their behaviour, or it could be the way in which the environment forces them to move. This latter condition is interesting because it forces correlated motion onto otherwise uncorrelated targets. This is how tracking targets such as vehicles (constrained by the road and other traffic) and blood cells (constrained by the blood vessels) can benefit from using an algorithm such as MPS, which should identify and make use of the information in such coordinated motions implicitly.

6.4. A final summary

As has been seen, the ability to monitor animals automatically, and hence reap the welfare and economic rewards that in turn can bring is a realistic goal with the current state of hardware technology and target tracking methods. Perhaps we are not at the stage yet where we can monitor multiple interacting targets with low or zero error rates for long periods, but the kind of success rates found in the tracking experiments for the MPS algorithm (Chapter 5) suggest the current state of the art could be a useful tool to aid the manual 'tracking' process often required in behavioural experiments. Chapter 3 illustrates that visual tracking technology can allow the automatic placements of sensors onto animals; however, the real cost of implementing this system is currently an unknown, and unless this kind of technology can be guaranteed to improve profits or is enforced by legislation, few stockman would be likely to adopt it.

Visual tracking, though, is a fast moving field which is offering more robust and competent algorithms each year. The MPS algorithm is one such contribution which offers a general framework in which coordinated motions can be used to guide tracking. Such developments bring the exciting goal of accurate, robust tracking of many targets closer to reality.

References

Allen,P.E. and Thorpe,C.E. Some approaches to finding birds in video imagery. CMU-RI-TR-91-34. 1991. The Robotics Institute, Carnegie Melon University.

Balch,T., Khan,Z. and Veloso,M. 2001. Automatically tracking and analyzing the behavior of live insect colonies. *5th International Conference on Autonomous Agents, AGENTS 2001, May 28th - June 1st, Quebec, Canada.* 521-528.

Bar-Shalom,Y., Fortmann,T.E. and Scheffe,M. 1980. Joint probabilistic data association for multiple targets in clutter. *Proc. Conf. on Information Sciences and Systems* 404-409.

Barron,J.L., Fleet,D.J. and Beauchemin,S.S. 1994. Performance of optical flow techniques. *International Journal of Computer Vision* **12,** 43-77.

Best,R.G. 1981. Infrared emissivity and radiant surface temperatures of Canada and Snow Geese. *Journal of Wildlife Management* **45,** 1026-1029.

Beynon,M.D., Van Hook,D.J., Seibert,M., Peacock,A. and Dudgeon,D. 2003. Detecting abandoned packages in a multi-camera video surveillance system. *IEEE Conference on Advanced Video and Signal Based Surveillance, July 21-22, 2003, Miami, Florida.* 221-228.

Black,M.J. and Jepson,A.D. 1998. Recognizing temporal trajectories using the Condensation algorithm. *Proc. International Conference on Automatic Face and Gesture Recognition, Nara, Japan, 1998.* 16-21.

Blake,A. and Isard,M. 1998. *Active Contours*. Second edn. Springer-Verlag.

Bose,B. and Grimson,E. 2003. Ground plane rectification by tracking moving objects. *Proc. Joint IEEE International Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance. Nice, France, October 2003* 9-16.

Brandl,N. Measuring pig travel by image analysis. http://nabilnabil.homestead.com/files/movtrack.htm . 2005. 16-2-2006.

Branson,K., Rabaud,V. and Belongie,S. 2003. Three brown mice: See how they run. *In Proc. of the Joint IEEE Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance at ICCV, 2003.* 78-85.

Brøndum,J., Egebo,M., Agerskov,C. and Busk,H. 1998. On-line pork carcass grading with the Autofom ultrasound system. *Animal Science* **76,** 1859-1868.

Bulpitt,A.J., Boyle,R.D. and Forbes,J.M. 2000. Monitoring behavior of individuals in crowded scenes. *Measuring Behavior 2000, 3rd International Conference on Methods and Techniques in Behavioural Research, 15-18 August 2000, Nijmegen, The Netherlands* 28-30.

Buxton,H. 2003. Learning and understanding dynamic scene activity: a review. *Image and Vision Computing* **21,** 125-136.

Canny,J. 1986. A computational approach to edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **8,** 679-698.

Chen,D., Yang,J. and Wactlar,H.D. 2004. Towards automatic analysis of social interaction patterns in a nursing home environment from video. *6th ACM SIGMM Int. Workshop on Multimedia Information Retrieval, in Proc. of ACM Multimedia 2004, October, New York* 283-290.

Chin,R.T. and Dyer,C.R. 1986. Model-based recognition in robot vision. *Computing Surveys* **18,** 67-108.

Chow,T.W.S. and Cho,S. 2002. Industrial neural vision system for underground railway station platform surveillance. *Advanced Engineering Informatics* **16,** 73-83.

Cohen,L.D. 1991. On Active Contour Models and balloons. *CVGIP: Image understanding* **53,** 211-218.

Collins,R.T., Lipton,A.J., Kanade,T., Fujiyoshi,H., Duggins,D., Tsin,Y., Tolliver,D., Enomoto,N., Hasegawa,O., Burt,P. and Wixson,L. A system for video surveillance and monitoring. Carnegie Mellon University report, CMU-RI-TR-00-12. 2000.

Conrady,A. 1919. Decentred lens-systems. *Monthly Notices of the Royal Astronomical Society* **79,** 384-390.

Cootes,T.F. and Taylor,C.J. 1992. Active Shape Models - 'Smart Snakes'. *Proc. British Machine Vision Conference, Leeds, UK* 266-275.

Cootes,T.F., Taylor,C.J., Cooper,D.H. and Graham,J. 1992. Training models of shape from sets of examples. *Proc. British Machine Vision Conference.* 1992*,* 9-18.

Couzin,I.D., Krause,J., Franks,N.R. and Levin,S.A. 2005. Effective leadership and decision making in animal groups on the move. *Nature* **433,** 513-516.

Cupillard,F., Brémond,F. and Thonnat,M. 2001. Tracking groups of people for video surveillance. *Proc. of the 2nd European Workshop on Advanced Video-Based Surveillance Systems, Kingston UK, September 4th 2001* 88-100.

Devernay,F. and Faugeras,O. 2001. Automatic calibration and removal of distortion from scenes of structured environments. *Machine Vision and Applications* **13,** 14-24.

Dick,A.R. and Brooks,M.J. 2003. Issues in automated visual surveillance. *International Conference on Digital Image Computing: Techniques and Applications. Sydney, December 2003* 195-205.

Doucet,A. On Sequential Simulation-Based Methods for Bayesian Filtering . CUED/F-INFENG/TR. 310. 1998. Cambridge University Department of Engineering.

ECM . Agroscan Homepage. http://www.agroscan.com/gb/index.htm . 2005. 16-2-2006

Farid,H. and Popescu,A.C. 2001. Blind Removal of Lens Distortion. *Journal of the Optical Society of America A* **18,** 2072-2078.

Forsyth,D.A. and Ponce,J. 2003. *Computer Vision: A Modern Approach*. Prentice-Hall.

French,A.P., Frost,A., Pridmore,T.P. and Tillett,R.D. 2003. An image analysis system to guide a sensor placement robot onto a feeding pig. *Proc. Irish Machine Vision and Image Processing Conference, 3rd-5th September 2003, Coleraine.* 185-192.

Frost,A., French,A.P., Tillett,R.D., Pridmore,T.P. and Welch,S.K. 2004. A vision guided robot for tracking a live, loosely constrained pig. *Computers and Electronics in Agriculture* **44,** 93-106.

Frost,A., Mottram,T.T., Street,M.J., Hall,R.C., Spencer,D.S. and Allen,C.J. 1993a. A field trial of a teatcup attachment robot for an automatic milking system. *Journal of Agricultural Engineering Research* **55,** 325-334.

Frost,A., Schofield,C.P., Beaulah,S.A., Mottram,T.T., Lines,J.A. and Wathes,C.M. 1997. A review of livestock monitoring and the need for integrated systems. *Computers and Electronics in Agriculture* **17,** 139-159.

Frost,A., Street,M.J. and Hall,R.C. 1993b. The development of a pneumatic robot for attaching a milking machine to a cow. *Mechatronics* **3,** 409-418.

Frost,A., Tillett,R.D. and Welch,S.K. 2000. The development and evaluation of image analysis procedures for guiding a livestock monitoring sensor placement robot. *Computers and Electronics in Agriculture* **28,** 229-242.

Gonzalez,R.C. and Woods,R.E. 1992. *Digital Image Processing*. Addison-Wesley Publishing Company.

Green, J., French, A. CamCap Video Processing Environment for Video Streams. http://www.cs.nott.ac.uk/~jzg/nottsvision/. Some components by other authors. 16-02-2006

Grimson,W.E.L., Stauffer,C., Romano,R. and Lee,L. 1998. Using adaptive tracking to classify and monitor activities in a site. *Proc. IEEE Computer Vision and Pattern Recognition* 22-31.

Heap,T. 1995. Real-time hand tracking and gesture recognition using smart snakes. *Proc. Interface to Human and Virtual Worlds, Montpellier, France, June 1995* 261-271.

Henderson,J.V. 1999. Flocking Behaviour of Ducks in Response to Predator Stimuli. University of Bristol.

Horn,B.K.P. and Schunck,B.G. 1981. Determining optical flow. *Artificial Intelligence* **17,** 185-203.

Intel . Open Source Computer Vision Library. http://www.intel.com/technology/computing/opencv/index.htm . 2005. 16-2-2006.

Intille,S.S. and Bobick,A. 2001. Recognizing planned, multiperson action. *Computer Vision and Image Understanding* **81,** 414-445.

Intille,S.S., Davis,J.W. and Bobick,A. 1997. Real-time closed-world tracking. *Proc. Conference on Computer Vision and Pattern Recognition, June 1997* 697-703.

Isard,M. and Blake,A. 1996. Contour tracking by stochastic propagation of conditional density. *ECCV, Cambridge, UK* **1,** 343-356.

Isard,M. and Blake,A. 1998a. A mixed-state CONDENSATION tracker with automatic model-switching. *Proc 6th Int. Conf. Computer Vision* 107-112.

Isard,M. and Blake,A. 1998b. Condensation - conditional density propagation for visual tracking. *International Journal of Computer Vision* **29,** 5-28.

Isard,M. and Blake,A. 1998c. ICONDENSATION: Unifying low-level and high-level tracking in a stocastic framework. *Proc 5th European Conf. Computer Vision* **1,** 893-908.

Ivanov,Y. and Bobick,A. 1999. Recognition of multi-agent interaction in video surveillance. *Proc. Int. Conf. Computer Vision* **1,** 169-176.

Ji,L. and Yan,H. 2002. Loop-free snakes for highly irregular object shapes. *Pattern Recognition Letters* **23,** 579-591.

Johnson,N. and Hogg,D. 1996. Learning the distribution of object trajectories for event recognition. *Image and Vision Computing* **14,** 609-615.

Kalman,R.E. 1960. A new approach to linear filtering and prediction problems. *Journal of Basic Engineering* **82,** 35-45.

Kass,M., Witkin,A. and Terzopolous,D. 1988. Snakes: Active contour models. *International Journal of Computer Vision* **1,** 321-331.

Khan,Z., Balch,T. and Dellaert,F. 2003. Efficient particle filter-based tracking of multiple interacting targets using an MRF-based motion model. *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems, Las Vegas. 27th-31st October 2003* **1,** 254-259.

Khan,Z., Balch,T. and Dellaert,F. 2004. An MCMC-based particle filter for tracking multiple interacting targets. *Proc. ECCV 2004, Prague, Czech Republic, May11th-14th, 2004.* 279-290.

Khan,Z., Balch,T. and Dellaert,F. 2005a. MCMC-based particle filter for tracking a variable number of interacting targets. *IEEE Transactions on Pattern Analysis and Machine Intelligence (in press)*.

Khan,Z., Balch,T. and Dellaert,F. 2005b. Multitarget tracking with split and merged measurements. *IEEE Computer Society Conference on Computer Vision and Pattern Recognition, San Diego, CA, 2005.* **1,** 605-610.

Liebowitz,D. and Zisserman,A. 1998. Metric rectification for perspective images of planes. *Proc. Conf. on Computer Vision and Pattern Recognition, 23rd-25th June 1998, Santa Barbara, USA* 482-488.

MacCormick,J. and Blake,A. 2000. A probabilistic exclusion principle for tracking multiple objects. *International Journal of Computer Vision* **39,** 57-71.

Magee,D.R. and Boyle,R.D. 1999. Feature tracking in real world scenes (or how to track a cow). *Proc. IEE Colloquium on Motion Analysis and Tracking, IEE, May 1999* 2-1-2/7.

Magee,D.R. and Boyle,R.D. 2000. Spatio-temporal modeling in the farmyward domain. *Proc. IAPR International Workshop on Articulated Motion and Deformable Objects* 83-95.

Magee,D.R. and Boyle,R.D. 2002. Detecting lameness in livestock using "Re-sampling condensation" and "Multi-stream cyclic hidden Markov models". *Image and Vision Computing* **20,** 581-594.

Makris,D. and Ellis,T. 2002. Path detection in video surveillance. *Image and Vision Computing* **20,** 895-903.

Marana,A.N., Velastin,S.A., Costa,L.F. and Lotufo,R.A. 1998. Automatic estimation of crowd density using texture. *Safety Science* **28,** 165-175.

Marchant,J.A. and Onyango,C.M. 1995. Fitting grey level point distribution models to animals in scenes. *Image and Vision Computing* **13,** 3-12.

Marchant,J.A. and Schofield,C.P. 1992. Extending the snake image processing algorithm for outlining pigs in scenes. *Computers and Electronics in Agriculture* **8,** 261-275.

Marchant,J.A., Schofield,C.P. and White,R.P. 1999. Pig growth and conformation monitoring using image analysis. *Animal Science* **68,** 141-150.

Marr,D. 1982. *Vision: A Computational Investigation into the Human Representation and Processing of Visual Information*. Freeman.

McInerney,T. and Terzopoulos,D. 1996. Deformable models in medical image analysis: a survey. *Medical Image Analysis* **1,** 91-108.

McKenna,S.J., Jabri,S., Duric,Z., Rosenfeld,A. and Wechsler,H. 2000a. Tracking Groups of People. *Computer Vision and Image Understanding* **80,** 42-56.

McKenna,S.J., Jabri,S., Duric,Z. and Wechsler,H. 2000b. Tracking interacting people. *Proc. International Conference on Automatic Face and Gesture Recognition* 348-353.

Meat and Livestock Commission 2004. *Pig Yearbook 2004*. Milton Keynes: BPEX, Winterhill House.

Needham,C.J. and Boyle,R.D. 2001. Tracking multiple sports players through occlusion, congestion and scale. *Proc. British Machine Vision Conference* 93-102.

Noldus,L.P.J.J., Spink,A.J. and Tegelenbosch,R.A.J. 2001. EthoVision: A versatile video tracking system for automation of behavioral experiments. *Behavior Research Methods, Instruments and Computers* **33,** 398-414.

Noldus,L.P.J.J., Spink,A.J. and Tegelenbosch,R.A.J. 2002. Computerised video tracking, movement analysis and behaviour recognition in insects. *Computers and Electronics in Agriculture* **35,** 201-227.

Nummiaro,K., Koller-Meier,E. and Van Gool,L. 2003. An adaptive color-based particle filter. *Image and Vision Computing* **21,** 99-110.

Okubo,A. 1986. Dynamical aspects of animal grouping: swarms, schools, flocks and herds. *Adv. Biophysics* **22,** 1-94.

Perrin,D.P. and Smith,C.E. 2001. Rethinking classical internal forces for active contour models. *Proc. IEEE Computer Vision and Pattern Recognition* **2,** 615-620.

Rayor,L.S. and Uetz,G.W. 1990. Trade-offs in foraging success and predation risk with spatial position in colonial spiders. *Behavioral Ecology and Sociobiology* **27,** 77-85.

Reid,D.B. 1979. An algorithm for tracking multiple targets. *IEEE Transactions on Automatic Control* **AC-24,** 843-854.

Remagnino,P., Baumberg,A., Grove,T., Hogg,D., Tan,T., Worrall,A. and Baker,K. 1997. An integrated traffic and pedestrian model-based vision system . *Proc. British Machine Vision Conference* **2,** 380-389.

Renco Corp. Renco Lean-meater. http://www.rencocorp.com/leanmeater.htm . 2004. 16-2-2006.

Reynolds,C.W. 1987. Flocks, herds and schools: a distributed behavioural model. *Computer Graphics* **21,** 25-34.

Roberts,L.G. 1965. Machine perception of 3-D solids. *Optical and Electro-optical Information Processing, MIT Press* 159-197.

Savory,C.J. 1995. Feather pecking and cannibalism. *World's Poultry Science Journal* **51,** 215-219.

Schofield,C.P. and Marchant,J.A. 1990. Image analysis for estimating the weight of live animals. *Proc. Optics in Agriculture, SPIE, Boston, MA, November 1990* 209-219.

Sergeant,D.M., Boyle,R.D. and Forbes,J.M. 1998. Computer visual tracking of poultry. *Computers and Electronics in Agriculture* **21,** 1-18.

Sidle,J.G. and Ziewitz,J.W. 1990. Use of aerial videography in wildlife habitat studies. *Wildl. Soc. Bull.* **18,** 56-62.

Sonka,M., Hlavac,V. and Boyle,R.D. 1993. *Image Processing, Analysis and Machine Vision*. 1st edn. Chapman and Hall.

Stanton,D., Bayon,V., Neale,H., Ghali,A., Benford,S., Cobb,S., Ingram,R., O'Malley,C., Wilson,J. and Pridmore,T. 2001. Classroom collaboration in the design of tangible interfaces for storytelling. *Proceedings of CHI'2001, Seattle, April 2001, ACM Press* 482-489.

Stotfold/MLC . Getting the best from your pigs - No. 4: Producing the right carcass. http://www.stotfoldpigs.co.uk/publish/pdfs/Pigs414.pdf 2001. Accessed: 16-2-2006. Operated by ADAS/IGER/University of Bristol.

Sumpter,N. 1999. The Robotic Sheepdog: Modelling Animal Behaviour From Image Sequences. PhD. School of Computer Studies, University of Leeds.

Sumpter,N., Boyle,R.D. and Tillett,R.D. 1997. Modelling collective animal behaviour using extended point distribution models. *Proc. British Machine Vision Conference* 242-251.

Swaminathan,R. and Nayar,S.K. 1998. Non-metric calibration of wide angle lenses. *Proceedings of the 1998 DARPA Image Understanding Workshop, Monterey, California (In these proceedings).*

Swedish Institute of Computer Science . Kidstory. http://www.sics.se/kidstory/ . 2005. Swedish Institute of Computer Science. 16-2-2006.

Tai,J.C., Tseng,S.T., Lin,C.P. and Song,K.T. 2004. Real-time image tracking for automatic traffic monitoring and enforcement applications. *Image and Vision Computing* **22,** 485-501.

Tillett,N.D. and Hague,T. 1999. Computer-vision-based hoe guidance for cereals - an initial trial. *Journal of Agricultural Engineering Research* **74,** 225-236.

Tillett,R.D., Frost,A. and Welch,S.K. 2002. Predicting sensor placement targets on pigs using image analysis. *Biosystems Engineering* **81,** 453-463.

Tillett,R.D., McFarlane,N., Wu,J., Schofield,C.P., Ju,X. and Siebert,J.P. 2004. Extracting morphological data from 3D images of pigs. *AgEng conference. Leuven, Belgium, 12th-16th Sept 2004.* 203-222.

Tillett,R.D., Onyango,C.M. and Marchant,J.A. 1997. Using model-based image processing to track animal movements. *Computers and Electronics in Agriculture* **17,** 249-261.

Trevelyan,J.P. 1989. Sensing and control for sheep-shearing robots. *IEEE Transactions on Robotics and Automation* **5,** 716-727.

Trevelyan,J.P. Univeristy of Western Australia's Robot Calibration Pages. http://www.mech.uwa.edu.au/jpt/CalibrationPages/MenuMain.htm . 2004. School of Mechanical Engineering. 16-2-2006.

Trucco,E. and Verri,A. 1998. *Introductory Techniques for 3-D Computer Vision*. Upper Saddle River, New Jersey: Prentice-Hall.

Tsai,R.Y. 1986. An efficient and accurate camera calibration technique for 3D machine vision. *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition CVPR'86, Miami Beach, Forida, 1986.* 364-374.

Tsutsumi,D. and Kita,Y. 2002. Motion tracking of cattle with a constrained deformable model. *Proc. 16th International Conference on Pattern Recognition* **1,** 372-376.

Uchida,K., Miura,J. and Shirai,Y. 2000. Tracking multiple pedestrians in crowd. *IAPR Workshop on Machine Vision Applications* 533-536.

Vaughan,R. 1999. Experiments in Animal-Interactive Robotics. Oriel College, Oxford.

Vaughan,R., Sumpter,N., Frost,A. and Cameron,S. 1998. Robot sheepdog project achieves automatic flock control. *Proc. Fifth International Conference on the Simulation of Adaptive Behaviour* 489-493.

Vaughan,R., Sumpter,N., Frost,A. and Cameron,S. 2000. Experiments in automatic flock control. *Robotics and Autonomous Systems* **31,** 109-117.

Vendenbroucke,N., Macaire,L. and Postaire,J.-G. 1998. Color pixels classification in an hybrid color space. *Proc. IEEE International Conference on Image Processing* **1,** 176-180.

Vincent,L. and Soille,P. 1991. Watersheds in digital spaces: An efficient algorithm based on immersion simulations. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **13,** 583-598.

Viscido,S.V., Miller,M. and Wethey,D.S. 2002. The dilemma of the selfish herd: the search for a realistic movement rule. *Journal of Theoretical Biology* **217,** 183-194.

Welch,G. and Bishop,G. An introduction to the Kalman Filter. Course 8. 2001. SIGGRAPH 2001.

Westphal,S.P. 2004. Big Brother keeps eye on lab animals. *New Scientist* **181,** 12-13.

Wouters,P., Geers,R., Parduyns,G., Goossens,K., Truyen,B., Goedseels,V. and Van der Stuyft,E. 1990. Image-analysis parameters as inputs for automatic environment temperature control in piglet houses. *Computers and Electronics in Agriculture* **5,** 233-246.

Wu,J., Tillett,R.D., McFarlane,N., Ju,X., Siebert,J.P. and Schofield,P. 2004. Extracting the three-dimensional shape of live pigs using stereo photogrammetry. *Computers and Electronics in Agriculture* **44,** 203-222.

Wyszecki,G. and Stiles,W.S. 1982. *Color Science: Concepts and Methods, Quantitative Data and Formulae*. Second edn. John Wiley and Sons.

Youssao,I., Verleyen,V., Michaux,C. and Leroy,P.L. 2002. Choice of probing site for estimation of carcass lean percentage in Pietrain pigs using real-time ultrasound. *Biotechnologie Agronomie Societe Et Environnement* **6,** 195-200.