Design and Performance Characterisation of a Modular Surveillance System for a Distributed Processing Platform

Mike Robinson

A thesis submitted for the degree of Doctor of Philosophy.
The University of Edinburgh.
September 26, 2001

Abstract

This thesis investigates pedestrian monitoring using image processing for state-of-the-art real-time distributed camera-processor architectures. An integrated design and evaluation process is proposed, where the surveillance task is analysed into component modules, each corresponding to a self-contained vision process. Different approaches to each process are implemented independently, using Object-oriented design principles to facilitate both system construction and module interchange during comparative testing.

Standard algorithms, from the computer vision literature, together with novel variants are used but the scope is restricted to what can be implemented to run in real-time, on a modest image processing engine. Comparison is made between median-based and mixture-of-Gaussian based methods for background representation, between connected component and boundary following approaches to object segmentation and between pixel-based and 2-D model based (PCA with cubic splines) methods for object classification.

Quantitative performance-characterization-data for existing solutions is not generally available, in the literature, in the form of bench-mark test-sequences and is time-consuming and costly to produce, for novel methods. A substantial test-data-set of real surveillance image-sequences has been acquired, to test the system and compare alternative approaches.

A novel performance-characterization technique is proposed: it offers comparative quantitative evaluation of the performance and resource requirements of a system. This approach is applied to the different system variants, comprised of the alternative module combinations.

The results of running the system variants on the test data are compared against manually derived ground-truth data for pedestrian detection. The performance characterization approach provided clear comparative data on performance and resource requirements for each variant, analysed by scene and event type. From a review of these results, the optimum module combination is chosen: this is a system composed of median-based background representation, boundary following object segmentation and model-based object classification.

Acknowledgements

I would like to thank my supervisors, Dr D Renshaw and Prof A F Murray for their invaluable help and advice throughout my studies. Also, I am indebted to my industrial sponsor, IndigoVision, and in particular my liaison there Dr. Mike Smart for their support.

Thanks are also due to my colleague Andrew Peacock, whose insight and assistance with programming issues were invaluable and who designed the framework and key classes for the department's Vision Systems programming library.

The Research was funded by EPSRC grant number 98000906 and CASE funding from IndigoVision.

Contents

List of figures

List of tables

Acronyms and abbreviations

.avi	audio video interleaved file extension
.dvi	digital video image file extension
.jpg	JPEG file extension
.m2v	MPEG-2 file extension
.pgm	portable grey map file extension
.ppm	portable pixel map file extension
AVS	Advanced Video-based Surveillance
BBN	Bayesian Belief Network
bps	bits per second
CamOS	IndigoVision's Camera Operating System
CCA	Connected Component Algorithm
CCTV	Closed Circuit Television
CIF	Common Intermediate Format, 352 x 288 (101,400 pixels)
DCT	Discrete Cosine Transform
DRAM	Dynamic Random Access Memory
DSP	Digital Signal Processor
EKF	Extended Kalman Filter
EM	Expectation-Maximisation
fps	frames per second
FPGA	Field Programmable Gate Array
Gb	Gigabyte(s)
HLIP	High Level Image Processing
HMM	Hidden Markov Model
IP	Internet Protocol
ITPMBVS	Integrated Traffic and Pedestrian Model-Based Vision System
JPEG	Joint Picture Experts Group
Kbps	Kilobits per second
LAN	Local Area Network
LLIP	Low Level Image Processing
Mbps	megabits per second

MPEG	Motion Picture Experts Group
MJPEG	Motion JPEG
Mb	Megabyte(s)
Mbps	Megabits per second
MLH	Maximum Likelihood Hypothesis
NTSC	National TV Standards Committee (analogue video standard)
OO	Object Oriented
PAL	Phase Alternating Line (analogue video standard)
PBMPLUS	Portable Bit Map Plus
PC	Personal Computer
PCA	Principal Components Analysis
PCB	Printed Circuit Board
PDF	Probability Density Function
PGM	Portable Grey Map
PPM	Portable Pix(el) Map
PSDB	Police Scientific Development Branch
PTZ	Pan-Tilt-Zoom
QCIF	Quarter Common Intermediate Format, 176 x 144 (25,300 pixels)
RGB	Red Green Blue (colour space representation scheme)
RAG	Region Adjacency Graph
RISC	Reduced Instruction Set Computer
ROC	Receiver Operating Characteristic
RSS	Remote Switch System
SIF	Standard Interchange Format
SOFM	Self-Organising Feature Map
SRAM	Static Random Access Memory
SVM	Support Vector Machine
TCP	Transmission Control Protocol
TTL	Time To Live
UID	Unique Identification Descriptor
UML	Unified Modelling Language
VPU	Video Processing Unit
VGA	Video Graphics Array, 640 x 480 (307,200 pixels)
VS	Vision Systems Research Group
YUV	luminance/chrominance (colour space representation scheme)

Chapter 1
Introduction

"The telescreen received and transmitted simultaneously. Any sound Winston made, above the level of a very low whisper, would be picked up by it; moreover so long as he remained within the field of vision which the metal plaque commanded, he could be seen as well as heard. There was of course no way of knowing whether you were being watched at any given moment." [3]

Figure 1.1: *Orwell's vision*

1.1 Introduction

1.1.1 Big Brother, the Panopticon and the surveillance society

In 1998, the author was called to jury service for a case involving assault, obtaining cash using threats, detaining individuals against their will and intimidation. The victims of the crime were reluctant in the extreme to identify the perpetrators because, as it emerged later, they had been threatened with retribution should they do so. There were however, witnesses to the crime which could not be intimated in this way and whose testimony served to convict the offender even without corroboration by the victims. These witnesses were a series of simple analogue surveillance cameras, both pole-mounted above a car park and positioned next to the lifts and stairwells of the crime scene.

Though this is an apparent success of and vindication for routine surveillance of public places, the positive nature of the outcome is actually somewhat less certain. The pictures presented were low quality, distant images and could not, in the opinion of the author, be confidently identified with the accused to the level of accuracy that would be demanded in **scientific** testing. Combined with the circumstantial evidence the match was close enough that, with the suggestive quality that images possess, a sufficient *feeling* of proof was given that a conviction was secured.

There is a dichotomy between the use of visual surveillance technology for the protection of society and the safeguarding of the rights of the individual. Whatever the uses to which this technology is put, however, it is likely that the higher the quality of the visual images and the more corroborative surveillance data that can be stored for cross-checking, the less chance there should be of errors. It is worth noting that research into recognition of unfamiliar faces suggests that people are generally very poor at recognising unfamiliar targets in the particular case of poor quality video [4].

If the video surveillance system used above had been complex enough to store a high resolution image of each individual's face to associate with the existing low resolution footage, a more secure conviction or alternatively a vindication of innocence could have been achieved. More than this, if the behaviour of the individuals could have been analysed in real-time, a response might have been made quickly enough to alleviate the suffering of the victims.

During the 1990s, the video surveillance industry in the UK experienced growth rates of over 10% annually ([5]) with installations both in company premises and several hundred city and town councils' public areas. The total value of the UK market was estimated in 1999 to be in excess of £343 million annually [6], with more than 300,000 cameras in shopping areas, housing estates, car parks and public facilities and an estimated 500 new cameras every week ([7]). The UK now has the largest number of surveillance cameras per capita of any country in the world.

The effectiveness of these cameras not just in detecting crime, but as deterrents, is a subject of controversy. Stirling council reported a 75% drop in crime in the three years subsequent to installing CCTV and Bournemouth a reduction in vandalism costs from £220,000 to £6,000 ([8]) after installing such cameras along its sea-front. However, a three-year study conducted by the Scottish Centre for Criminology found that recorded crime rose by 9% after introduction

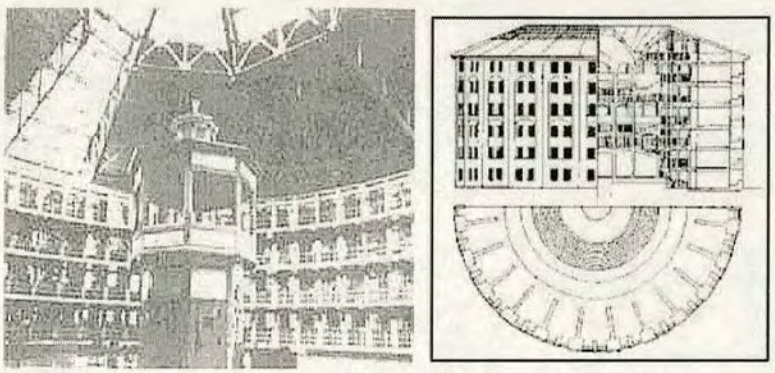

Figure 1.2: *Jeremy Bentham's Panopticon prison*

of cameras, while detected crime (leading to identification of a suspect) fell by 4% ([9]). They suggested that the amount of Home Office budget currently expended on the technology was out of proportion with the ambiguous outcome.

As well as dispute over the effectiveness of visual surveillance, concerns as to the ethical position have been presented by organisations such as Liberty. The natural rallying call has been claims of the onset a dystopian society reminiscent of that depicted in Orwell's '1984', where "Big Brother is Watching You" through the contemporary equivalent of the telescreen. Foucault has presented an alternative analysis of the effects of surveillance, referring to Jeremy Bentham's plans for the Panopticon prison in 1791 (figure 1.2). This presents a situation where all prisoners are segregated into cells around a central tower and where the guards can see without being seen, a critique intended to be a metaphor of the surveillance society and intended:

> *" ... to induce in the inmate a state of conscious and permanent visibility that assures the automatic functioning of power. So to arrange things that surveillance is permanent in its effects, even if it is discontinuous in its action ... "* [10]

Civil rights campaigners proclaim that the systematic attention of continual surveillance overturns the presumption of innocence, creating a situation where all actions are suspect and monitored to ensure compliance with the rules. Oppressive psychological effects are claimed to ensue, whereby persons under observation are pressured into conforming to the most conservative interpretation of acceptable behaviour. The routine monitoring of the workplace by employers has been cited as a particular example of the systematic over-use of the technology.

With current advances leading toward the ability to explicitly define 'unusual behaviour' and to have systems generate an automatic alarm when it is detected, the potential for abuse by the establishment and private interests can only increase. While such ethical considerations should not be neglected by the scientist or engineer, it is hard not to fall back on the claim that it is not the technology which causes the problem, it is the use to which the technology is put.

It is important in this area, more than most others, to keep in mind the very direct political and social impact of developments in the field when considering design and evaluation of new systems. It is important that the development of legal guidelines and protection follows close behind the increasing technological capabilities of advanced visual surveillance.

1.1.2 The history of surveillance

The importance of surveillance information is a constant which dates back beyond the dawn of civilisation. Before what would be man was significantly differentiated from the animal kingdom, an attribute needed in 'survival of the fittest' was an accurate and timely awareness of the surroundings both of the individual and the group. Predators and prey had to be detected as efficiently as possible and alarms transmitted to the group to initiate appropriate action. In every civilisation and for the whole of recorded history, the posting of guards at the perimeters of settlements and encampments has been vital to the protection of the group. The more capable these sentries and the better organised their communication with each other and their commanders, the more secure the area under their protection.

The importance of the rapid communication of surveillance data is portrayed in myth and legend, with the examples of the first Marathon run and Paul Revere's ride as evocative examples of their kind. A message of the perils of inadequate surveillance, lax security and poor intelligence is easy to extract from the tale of the Trojan Horse.

With the gradual but accelerating pace of technological innovation, the patterns of the surveillance methodology began to change incrementally. First the use of mirrors for signalling, then the invention of the telescopic lens, in the 13^{th} century, extended the range of human surveillance capabilities. The first application of camera technology to surveillance dates back at least as far as 1870, when the French military used aerial photographs captured from a hot air balloon to monitor troop movements.

Much later, in the 'electronic age', the telegraph, burglar alarm, police box, field radio and

walkie-talkie all brought ways to enhance communication between sentinels and command centres. The first prototype video camera-like product was invented in 1923 but, as with many fields, development in the area accelerated significantly significantly during World War II.

From the 1950s onwards, Closed Circuit TV (CCTV) cameras began to be employed for large, private surveillance tasks. Prior to this time, the only practical option for surveillance was one or more guards physically traversing the area under observation and manually viewing the situation. The communication of alerts had been improved by using electronic devices and alarms could be set to assist in detection, but essentially the process was one that would have been recognised by an ancient Roman centurion.

1.1.3 First-generation video-based surveillance

The surveillance model employed in the first commercial systems is one that did not significantly change for several decades. At its most basic, CCTV is a camera connected to a monitor for viewing an area remotely. A collection of such cameras would be placed around the zone of interest which could be an industrial complex, a military base, or some other secure premises. The cameras would be wired to a central control room by dedicated cabling and the images displayed on a corresponding number of monitors at that location.

Incremental advances in the technology have allowed more efficient usage of hardware and alternative viewing possibilities. Automatic camera switches have enabled auto-cycling, where the image at the monitor is replaced every few seconds by one from a different camera. Also, screens can be split to carry several camera outputs simultaneously, with an operator override where one image of interest can be singled out.

These advances primarily focused on reducing the hardware requirements of the system: installing CCTV represented a large capital outlay, although the extent of this reduced over time as components became cheaper. Assisting the operator was for many years restricted to issues of control room design, with viewing ergonomics investigated by, among others, the Police Scientific Development Branch (PSDB) in the UK. PSDB work suggested that 1 to 5 monitors per operator was the most efficient, with a primary monitor directly in front of the operator for incident detection. Secondary monitor banks could give an overview, and if placed just in the range of the peripheral vision, movement detection could be used as a cue for follow up. The use of auto-cycling and screen splitting was advised against in high risk surveillance as the first

negated the use of motion detection and introduced temporal 'blind spots', while the second produced unacceptable deterioration in image clarity.

In this traditional CCTV-augmented surveillance, all analysis would by necessity be carried out by the operator, requiring long periods of concentration, split over a potentially very large number of screens simultaneously. The ability of the human brain to perform complex visual analysis tasks with little apparent effort can not be overstated but, as with all tasks, performance tends to reduce over time. The number of operators could be increased, but only subject to remaining within the bounds of economic feasibility . *"Who watches the watchers"* is a pertinent query with respect to maintaining operator efficiency in this context.

Vigilance or *sustained attention* refers to "the ability of observers to maintain their focus of attention and to remain alert to stimuli for prolonged periods of time" [11]. This was first studied [12] at the request of the Royal Air Force after World War II to investigate the behaviour of radar operators watching for German U-boats. The study found that the accuracy of signal detections declined by about 10 percent after 30 minutes and continued to decline more slowly over a 2-hour session. A key finding of subsequent work was that this decline occurs in any situation which involves the need to look for a relatively infrequent stimulus over a continuous period of time. Of further relevance to assessing the unaided human monitoring paradigm, studies have shown [13] that task performance in an extended overnight period (including the period 23:00 to 06:30) drops as low as 30% to 40% below that seen during the daytime when the operator was well-rested.

The goal of first-generation video-based surveillance systems, to present operators with the maximum information about the monitored environment, is then constrained by the major limitation of the operator's attentiveness. Advanced Video-based Surveillance (AVS) considers second-generation systems which aim to take advantage of the further enhancements available from digital computing and communications techniques.

1.1.4 Computer vision and surveillance

The idea of feeding visual data to a computer for intelligent analysis dates back at least as far as 1966 when Marvin Minsky set just such a task for a first year undergraduate. The complexity of the task was underestimated then and has been one of the iconic goals of Artificial Intelligence up to the end of the millennium and beyond.

A complete computational theory of vision was not fully developed until the work of Marr [14] in the late 1970s which focused on the processes undertaken in the primate visual cortex. He suggested that full description of an observed scene was achieved by constructing a number of distinct intermediate representations. First a description of all light intensity discontinuities constituted the *raw primal sketch*, followed by grouping procedures to find *primitives* (boundaries and regions) for the *full primal sketch*. The culmination of early visual processing adds analysis of depth, motion and shading to build the *2.5D sketch*, a viewer centred representation of the surrounding environment. He proposed that the higher level process of object recognition was carried out by matching against stored 3D models of each object, constructed from basic 'LEGO brick' elements. Although a useful for defining terms and identifying issues, this sole complete framework is now generally considered insufficient as a context for advancing computational vision research.

Vision research has progressed significantly in many areas corresponding to elements of Marr's model, including approaches to finding many kinds of primitives (edges, corners, lines etc.), automatic segmentation of regions from still images, stereoscopic depth extraction, motion analysis and shape-from-shading. The design and implementation of a complete system to achieve a human-equivalent understanding of a scene from visual information is still a remote goal however. Chapter 5 includes a summary of some recent analyses of the lack of a solid characterisation procedure as one constraint on progress. The ideal for such an approach with practical and commercial application can be stated as a system which, after an initial installation, can be simply given a high level command such as *" Observe the scene and report any interesting behaviour in regions X, Y and Z"*. The system would then use the resources available in the most efficient manner to achieve the desired ends.

A significant obstacle to the understanding of how complicated the task of visual processing and understanding is, is the apparent ease with which we as observers achieve the task ourselves. Humans are able to accomplish the processing of signals (visual, acoustic, tactile and olfactory) under a wide range of environmental conditions with little or no conscious effort. The very conceptualising of how hard the problem may be is difficult when we can observe a two-year old child managing the task that we wish to achieve technologically. The initial excitement generated by the development of *neural nets* as ways to potentially model the workings of the brain, while not promptly leading to an Asimov robot, served to highlight the complexity of the process in computational terms. The current goal of artificial brain construction has reached

the level of aiming to achieve a level of complexity equal to that of a kitten's brain [15].

Both decomposition of the problem into sub-processes reminiscent of the Marr model and constraining the vision analysis requirement to more restricted domains are common approaches to making the task more tractable. Consider the surveillance example of relevance to the investigation reported in this thesis, specifically the interpretation of the behaviour of pedestrians: a split into high, low and mid level processes (see section 2.1) is the usual first stage of differentiation. Research in the area, then, commonly considers the problem as comprised of distinct subprocesses, chosen according to the intuitive decomposition of the task (again, see section 2.1).

The ultimate goal of pure research in the area can still be considered to be Minsky's desire for a fully automated system for understanding the presented scenes. Systems which implement very restricted versions of this ability have reached the stage where they have been tested in real surveillance environments (see section 2.9). Such systems are currently able to deal with recognising a handful of pre-specified behaviours and this only after prolonged learning or manual calibration phases.

At the current stage of development, the optimum use of available image processing and understanding technologies is not then to *replace* human operators, but to *augment* their abilities and overall performance. As noted in section 1.1.3, a limiting factor in the ability of human centred systems is the degradation in their vigilance over prolonged periods of time. A restatement of a key goal of AVS then, could be the use of applicable computer vision processing techniques to assist in directing the attention of human operators, leaving them able to concentrate on critical high-level interpretation and decision making.

1.1.5 Industry resistance to new technology

A wide gap exists between the highly complex prototype systems developed in the academic community and the systems commonly employed in industrial contexts. To explain the slow uptake of new technology, Pavlidis et al [16] cite some significant characteristics of the security industry.

- **Low Profit Margin** is cited as the primary restricting factor: with a constant struggle just to break even in the profit and loss account and a strategic planning horizon commonly

restricted to six months, anything beyond the provision of the basic acceptable service will generate considerable additional business risk. Security is commonly a low priority in commercial budgeting, except for brief periods after a security lapse, and is considered to add nothing to the (financially) important numbers.

- **Resistance to change and low-tech culture** are both ascribed as due to the nature of the managers in the industry. Their training and experience is usually from within the business after an introduction as operatives and they are more comfortable with the low tech status quo. An atmosphere of suspicion of higher technical innovation generally is exacerbated by the fear of consequential job losses.

- **Hardware bias** is reflected in the 'tunnel vision' seen in many camera manufacturers, the principal proponents of technological advance in the industry. Although pushing the development of hardware solutions, the out-fitting of these cameras with appropriate software is commonly neglected.

In addition to these points highlighted in that paper, there is a general cynicism held by many industry decision makers in respect of the claims made by academic researchers.

1.1.6 Second-generation video-based surveillance

As discussed in section 1.1.3, surveillance systems have until recently been developed according to a single basic architectural model: a collection of cameras placed around an area to be monitored, connected to a central control location using dedicated, hard-wired links. Transmission of images would be along copper cables (although latterly with the option of optical fibre) using standard analogue image formats, often of quite low quality. Such a security system was a major investment for a company both in monetary terms and in terms of the hours of set-up time and inconvenience of installation.

An advent which mitigates some of these considerations is the development of digital systems which can interface with an entity's existing computer network structure. Although digital systems have been available to the CCTV industry for more than a decade, there has been considerable reluctance to accept such a radical change in the technology basis. This has been due in part to a lack of understanding of the technology and the benefits it offers as well as general suspicion of any such a relatively new developments. Consequently, a large shift towards

the use of digital platforms has only begun over the last three years, despite the considerable benefits it offers.

Each camera is connected to a local hardware package that enables both interface into the network via a standard Ethernet link and the option of local processing. In its most basic operation, such a system allows the transmission of compressed video streams along the network to be decoded and decompressed at the central control location. Using this architecture, the initial capital outlay can be much reduced and the inconvenience of laying dedicated cabling can be almost entirely eliminated. The systems are much more flexible and adaptable as new cameras can be installed and existing ones moved with corresponding ease.

A potential downside of the basic operation mode noted above is the continuous use of the finite transmission bandwidth resources of the network by default transmission of all video data. Furthermore, the amount of data to be processed at the central location (usually by human operators, as noted above) is unchanged, although the picture quality may be significantly enhanced.

Both problems can be addressed by the next logical step in the approach: remote processing of video data. If the on-camera processing is extended beyond digitisation, compression and transmission to allow some degree of image analysis, video data can be transmitted only when pre-specified alarm criteria are met. Resource availability on-camera will inevitably be somewhat restricted due to the physical limitations on the hardware. It will be necessary to develop/adapt existing methods to suit such a digital camera based network so it uses the most efficient approach available for the task at hand.

1.1.7 Evaluation of computer vision processing techniques

A common feature of most of the full analysis systems described in Chapter 2 is the use of complex models in predicting/evaluating object behaviours. Considering the Integrated Traffic and Pedestrian Model-Based Vision System (section 2.3) in particular, the steps required include simple frame differencing, boundary extraction, fitting of a spline and Principal Components Analysis (PCA) to build models off-line. These models must then be used by a Kalman filter on live footage (again processed by frame differencing, boundary extraction and fitting of a spline) for the matching/tracking of objects.

While the use of such a complex modelling approach can provide good results and may be im-

portant in enhancing their quality and reliability, they do impose a considerable computational overhead. Although the cost of computational power continues to fall, in practical applications it is wise to remain prudent with resources, most especially in a commercial context. In some situations (for example safety critical applications) a cost versus benefit evaluation of the approach would be most likely to endorse the use of such an approach whereas in others (for example low level security) a simpler, less resource-hungry method may be more suitable.

The first step in performing such a cost versus benefit analysis is to derive a clear, quantitative assessment of the costs (such as cpu time and memory requirement, hardware and software costs, installation and maintenance) and benefits (such as positive/negative failure rates, speed of operation, complex scene handling, flexibility) for representative approaches. In a system where processing may be limited to that possible at a relatively low-specification processor, the choice between excellent results with prohibitive resource requirements and adequate results with feasible demands is very significant. Such analysis is not common within the computer vision literature and the data to use for choosing optimal solutions is not generally available.

It is not efficient to design a system for practical usage, to implement it on the desired platform and to then conduct an analysis of the performance and resource requirements. At this stage it is likely be too late to conduct significant revisions, at least until the next release of the system should there be one. To develop a suitable system, some consideration of the performance characterisation issues should be taken into account from the very beginning of the design phase.

Following the principles of Marr's theory, the general pattern in vision research and the principles of object oriented design, it is best to consider the system in terms of the component processes. If these components are chosen to correspond to the usual decomposition of the problem (section 2.1), they will also correspond to sub-problems for which published work exists in the vision literature. Any published results from previous work will then assist in the selection of the approaches to adopt in the construction of a new system.

Ideally, the results available on performance of extant systems, combined with detailed performance characterisation of each individual new algorithm could be used to nominate the best overall construction of a system to work on a specified platform. However, as discussed in Chapter 5, detailed performance characterisation results are not generally available in the literature. Furthermore, the resources required to obtain such results from each new module are

restrictive within the context both of research projects and most commercial developments. An alternative approach to such a design problem is needed.

Any evaluation requires a context in terms of the anticipated usage and the platform upon which the system is to run. In the case to be considered, the usage is as a surveillance system, primarily designed to detect pedestrians in scenes. This task was chosen as it is one of the smallest self-contained practical applications constituted from discrete sequential vision processes. Using this as the *specified application* will simplify the evaluation task in this pilot implementation. The platform on which the system is to run is a version of the remote processing based system described in section 1.1.6. The procedures are to run at the remote processor without any initial recourse to central processing facilities. The platform details are described in full in Chapter 3.

As discussed in full in Chapter 4, the chosen approach for system evaluation is to apply Object Oriented design principles to the problem to develop a modular solution to the surveillance problem under consideration. Each module will correspond to one of the sub-problems noted above and will initially be implemented separately. Using the available results in the literature for pre-selection, a range of algorithm solutions will be implemented for each module.

Performance characterisation will not be at the module level, but will be upon the combinations of modules comprising alternative *system variants*. Testing will be on real surveillance sequences which embody a range of environmental and behavioural attributes. These scenes will be stored for use by each *system variant* and will be made available to third parties for independent testing. Performance against manually acquired ground truth and resource requirement measures will be combined to give an overall evaluation for the suitability for the final platform.

Full per-module performance characterisation including failure modes and error propagation would offer greater generalisation and usefulness to third parties, but this must be balanced against the resources such comprehensive testing demands.

1.2 Motivation and contributions

The motivation behind this project has two principal aspects.

First, a review of the literature underlying the development of new surveillance solutions (see chapter 2) indicates that a gap exists between the needs of the surveillance industry and the results generally available in that literature. The industry looks for a precise quantitative char-

acterisation of the abilities, limitations and error profiles of vision processing units to assist in constructing systems to meet customer specifications. Vision research primarily focuses on pushing the boundaries of system performance and exploring interesting technical avenues to investigate novel approaches. Results are commonly presented in image-like formats or 'demos' and/or using limited test data.

Unless and until the two communities become more aligned, it would be of great value to provide a framework within which a compromise solution is possible. A principal contribution of this project is to develop and evaluate a methodology intended to constitute such a framework.

Second, the original problem in relation to which the need for such a methodology became apparent was the design and characterisation of a surveillance application to run on a state-of-the-art, distributed processing smart camera network. Applying the methodology to this problem should provide a recommendation of the best selection between candidate approaches to achieving the surveillance goal.

A further contribution is that a qualitative evaluation of the comparative performances of the individual approaches implemented in the system will also be obtained. The final principal contribution is the beginning of a data base of real surveillance test sequences, available to third parties for use in future design problems.

1.3 Thesis goals

> *This thesis proposes that an integrated design and characterisation methodology for the construction of vision systems for a specified application, based on object oriented design principles, can efficiently provide a recommendation as to an appropriate implementation. Such a methodology can also contribute to an evolving database of test sequences and comparative performance information of more general use to third parties.*

The overall goals of this project then, are as follows:

- To survey existing work in the fields of vision system performance characterisation and those areas of vision research relevant to the application to be designed.

- The creation and evaluation of a new methodology for the integrated design and characterisation of an appropriate practical solution to a specified vision processing problem.
- Using this methodology, to design a modular surveillance application to run on a distributed processing architecture and to use object oriented design principles to implement a selection of standard algorithms and novel approaches.
- To characterise the performance of the different *system variants* which these modules constitute on real surveillance image sequences.
- To compare these performances against manually acquired ground truth and review resource requirements to achieve a relative analysis of the performance of the *candidate solutions*.
- To initialise a data base of real surveillance test sequences to be available to third parties.
- To draw conclusions from the investigations and recommend an appropriate *system variant* to migrate to the intended platform plus improvements to the design methodology.

1.4 Thesis structure

Chapter 2 is a survey of existing work in the fields of vision system performance characterisation and those areas of vision research relevant to the application to be designed.

Chapter 3 describes the particular platform upon which the final system variant is to run and presents the general design considerations for the implementation phase. This, in parallel with consideration of the literature in Chapter 2 provides the specifications of the application under design.

Chapter 4 goes into the detail of the *candidate solutions* to be implemented for evaluation towards designing the most appropriate *system variant*.

Chapter 5 discusses the position in respect of performance characterisation in the vision community in the past, recent new initiatives to the process and describes the motivation for the current approach.

Chapter 6 presents the results of the performance characterisation tests, offers explanations for unanticipated patterns and draws initial conclusions as to the effectiveness of the *candidate*

solutions.

Chapter 7 draws conclusions from the work as to the most appropriate choice for the *system variant* to migrate to the intended platform and an evaluation of the combined performance characterisation/design process. It also discusses possible improvements for each, limitations to the current versions and suggests avenues for future work.

Chapter 2
Background

2.1 Introduction

There are two main areas which must be investigated to provide an adequate background for the work undertaken in this project. The first is the relatively small body of work which has been undertaken in the field of performance characterisation as applied to vision research, considered in section 2.2. The balance of the chapter considers the research into approaches to the task of visual surveillance, primarily the automated surveillance of pedestrians (see section 3.3).

To provide a structure for the investigation of the surveillance topics, it is useful to consider the processes involved in the interpretation of the behaviour of pedestrians in a scene split into three levels (as in [17]):

- Low level processes which implement algorithms primarily at the pixel level to produce primitives such as edge strength maps, corner maps and segmented object maps, which are still primarily within the image-like domain.
- High level processes such as semantic behaviour analysis, 3D shape inference and identity hypothesising which simulate intelligent processing of the data.
- Mid level processes such as model fitting, tracking and object classification which bridge the gap between the other two levels.

Most of the design, implementation and evaluation approaches which constitute the bulk of this work are focused on low and mid level processing but it is also important to consider some developments in high level analysis. This will provide a context for the current work and suggest some of the possibilities for future extension.

There is a substantial amount of research (see sections 2.7 and 2.8) in the area of pedestrian detection and tracking, and systems for achieving this task are almost universally considered as constituted from the following distinct subprocesses:

1. Initialisation and camera calibration
2. Image capture
3. *(Optional)* Image preprocessing
4. Background estimation
5. Foreground segmentation
6. Object growing and labelling
7. Object classification
8. Object tracking
9. *(Optional)* Higher level processing

Apart from the section relating to performance characterisation (see above), the investigations in this chapter will be structured in line with this decomposition. Research endeavours which only extend to a certain subprocess will be considered in the section relating to that level. The approaches applied in earlier subprocess steps will be considered and discussed within that context. In each section, methods will be grouped according to the similarity of the approaches employed at the highest subprocess level.

In addition to the overall system design and characterisation approach, a principal area for novel work within the current context is in investigation of the performance of model-based techniques for detection and tracking as compared to less computationally intensive approaches in deducing object behaviour. An additional area for interesting qualitative comparison exists between the traditional long-term average based uni-modal background representation approaches and the more complex multi-modal options.

One key approach, which provided motivation for several of the module *candidate solutions* is described first (section 2.3), followed by systems which implement some or all of the subprocesses of interest. The investigated systems are presented ordered in terms of the ultimate processing step investigated in the published work, in the terms of the above noted subprocesses.

2.2 Performance characterisation in computer vision

The approaches emerging to correct the deficiency in empirical testing have several strands, all of which have contributions to offer to the advancement of the field.

Bowyer [18] considers the experimental evaluation of individual vision algorithms (primarily edge detectors) using real images. He notes that substantial research effort is required in constructing image sets (input data), developing meaningful evaluation metrics, ascertaining ground truth and constructing supporting software tools. In his examples a wide range of evaluation processes are employed, including human visual ratings, piping output to an independent and invariant *structure from motion* algorithm and the use of *Receiver Operating (ROC) curves* (see section 5.5.6). Emphasis is made that the latter two approaches require *"truly enormous amounts of computer time"*.

As referred to below in section 5.2.2, Courtney [19] uses comparisons with other fields to derive six axioms for developing working vision systems:

1. Encourage modularity, emphasising reuse of existing modules to spread evaluation costs.
2. Define module characteristics using data sheets to give generic expressions of usage parameters.
3. Develop standard design guidelines, test environments and evaluation tools.
4. Give quantified 'safety margins' in design specifications.
5. Investigate and record system failure modes.
6. Investigate and record limits of use and build in reactions to these being exceeded.

Courtney and Thacker [20] discuss some of the issues involved in good practice for algorithmic testing. They note that the amount of data is a critical issue, with robust algorithms requiring *"huge quantities of data"* : for 99% reliability, hundreds of test images may be required. The range of the test data is very important too and should be arranged to be representative of the range over which the algorithm is intended to function. They note that test metrics should be quantitative and objective and that it is useful to associate them with an algorithm's failure modes. Again, ROC curves are proposed as useful metrics for feature detection algorithms.

In the second part of their paper, they discuss algorithm design principles and discuss how an understanding of the statistical properties of data and algorithmic operations is important. They emphasise that a thorough understanding of each algorithm's theoretical basis is important in constructing a stable system where errors propagate in a known and manageable form.

In [21], the FERET program, designed to set up a large database of facial images and in parallel a testing procedure to evaluate face recognition systems is presented. It is specified that the goal of the project is not to measure the effectiveness of performance of individual modules but to assess the performance of each system as a whole on a large data base. The central issue is that fully comparative evaluation is only possible using a common test database. In this example the construction of the database is performed independently of the systems' developers, as are the actual test procedures.

It is worth noting that although the database did cover a range of variables including different backgrounds, expressions, hair styles, augmentations (hats, glasses) ages etc. it focused on Caucasian subjects. In this respect the global usefulness of the study is somewhat impaired.

Clark and Courtney [22] too state that the best approach to characterising performance in the field as things stand, is the application of candidate vision techniques to a common set of data and go on to conduct a survey of the use of databases in this context (the ECVnet Survey). They emphasise that the databases must be representative of the problem in terms of overall size and the constituencies sampled. They also state that experimental design is of key importance, again recommending the use of ROC curves when assessing techniques.

They are careful to note that simple objective measures do not necessarily measure a useful quantity and so it is important that such metrics be chosen carefully.

In [23], Marik brings out the point that even a 'simple' computer vision task generates a complex algorithm-wise performance analysis problem. He draws on an analogy to a production line process comprised of multiple processing stages. From this idea he suggests the application of the theory of *Quality* from manufacturing analysis as a top-level framework for performance characterisation. This introduces the concepts of *true product performance* which is only measurable by the end user 'in the field' and *substitute performance* which is what is assessed during pre-release testing. A framework is presented where the correspondence between the two performances can be analysed at system and algorithm levels.

An evaluation strategy which has been developed specifically for surveillance applications is presented in [24]. The general surveillance problem is split into Image Processing, working with images to give a set of metrics and Image Understanding, which performs interpretation on the IP output to give semantically meaningful information. Image processing is further subdivided into Low Level Image Processing (LLIP) which attempts application-oriented filtering of noisy sequences plus a background representation and High Level Image Processing (HLIP) to give synthetic descriptions of image content.

The evaluation approach focuses on HLIP, which is subdivided into *blob detection*, *feature detection*, *blob tracking* and *feature tracking* sub-modules, all of which are parametrised. The optimisation problem for HLIP overall is considered as a search for the parameter configuration which will provide the best overall results. A formal definition of the optimisation criterion is given for the *blob detection* task in terms of comparison of results with ground truth evaluated by a human operator.

A distance measure between the ground truth and observed results is defined proportional to false alarm and missed detection probabilities as evaluated from the amount of overlap between images segmented automatically and images segmented by humans. The two distance components are plotted over 500 frames for a limited range of the four parameters which effect *blob detection*, using discrete steps 'depending on the parameter itself'. The result allows the choice of best parameter selection for optimising false alarm/missed detection independently and allows a trade-off to be made in parameter choice between the two metrics.

2.3 The 'Integrated Traffic and Pedestrian Model-Based Vision System'

A full surveillance system which has shown great abilities in the current problem domain, that of pedestrian detection, tracking and behaviour analysis is the Integrated Traffic and Pedestrian Model-Based Vision System (ITPMBVS) [25]. The system uses two integrated subsystems to monitor a car park scene, one to locate rigid objects (primarily cars) using a 3-D model and the other to track flexible objects (primarily people) using a 2-D model. Pedestrians are modelled using a flexible 2D model which is automatically learned using image sequences containing representative shapes [26] [27]. The latter inspired several component modules in the system under design and so the relevant components are described in detail in Chapter 4, but a summary

review is included here.

The background of the scene is represented using a time-averaged median approximation [28] (see section 4.4.2), with moving objects segmented by frame differencing between this and the current frame. After blurring and thresholding to remove noise, the boundary of each object is represented using a cubic B-spline, upon a sequence of which boundaries Principal Components Analysis (PCA) is performed to give the characteristic *eigenvalues* and *eigenvectors*. These correspond to the characteristic shape variations of the pedestrian motion and the model is constituted of the mean shape from the sequence and a subset of these eigenvalues and vectors.

This model is matched against moving objects in new scenes to classify as pedestrian/not-pedestrian. Each pedestrian object is specified by a group of parameters: its centroid, scale, orientation and a set of weighting parameters for the eigenshapes of the model. The parameters are used to correlate each object between frames with tracking implemented using a Kalman filter [29] [30] [31].

An object's trajectory is described using a series of flow vectors. A competitive learning network is used to model Probability Density Functions for these vectors. In this network, the trajectory points constitute input to the system where each image point is associated with a particular node. Using the learning rules with this input, a model is built up automatically over time which specifies typical trajectories for a scene against which new trajectories can be classified as typical or atypical [32] [33]. This representation includes information on interaction with mapped stationary objects (e.g. cars) and is amenable to construction of a spatiotemporal model for prediction [34].

2.4 Camera and platform issues

Overall hardware, system design and platform specific considerations are important in designing an approach which can take advantage of their full potential and recognise their limitations. As hardware and platform abilities and complexity increase, the impact on system design and control feedback is likely to become ever more significant.

2.4.1 Passive camera systems

Calibration is necessary to relate camera data to the real world using a projective transformation of world points to image points. Several of the systems considered require scene knowledge to be incorporated into the model to assist control, interpretation and other high level processes and manual calibration of the camera is currently a required step.

Work in the area has progressed to the level where it is possible to obtain calibration with no reference object for the increasingly difficult cases of pure translation with camera intrinsic parameters variable (e.g. zooming) [35], pure rotation with invariant camera intrinsic parameters [36] and pure rotation with variable camera intrinsic parameters [37].

As such camera calibration is considered as a preprocessing step and will not be implemented as a module in the system under consideration, details of the approaches used will not be presented here.

2.4.2 Multiple camera issues

In a distributed camera network it is advantageous to be able to combine information from different cameras to track an object as it moves from the view of one to that of another. Looking at the case of two cameras specifically, [38] presents a method whereby moving objects can themselves provide the information to enable this.

First, moving objects are segmented by subtraction of an adaptive background model, and their centroids are calculated and tracked over multiple frames. The two cameras each then have a list corresponding to a moving object's trajectory from their viewpoint. A estimation of a planar mapping between the two is given by a *least median of squares* measure and the coplanar points take the form of a homography from which the rough alignment of the ground plane can be calculated. This alignment given is sufficient to apply robust estimation techniques for planar alignment to give the fine alignment for the ground plane. Using the tracked object coordinates again, the epipolar geometry (the cameras' relative translation) can be calculated to finally align the data.

In [39], the point is made that more is needed from surveillance solutions to assist the operator by reducing the 'flood of images' generated by multiple cameras. The paper notes that operators in metro station control centres not only need to monitor multiple alternating im-

ages but can have a multitude of secondary tasks assigned. Intelligent surveillance systems, which themselves act like an operator by giving pre-selection of possibly interesting images are recommended as the solution.

2.4.3 Current 'Remote Intelligent Surveillance Camera Systems'

An example of a current application of a remote intelligent surveillance camera system is given in [40], in operation at the Port of Savona in Italy. 24 cameras with PC board-based remote processing units (*Video Servers*) are sited over 65 hectares, linked to a central location using the port's existing telecommunication and information network. The video servers are interfaced using standard ethernet connectors and transmit data in compressed form using TCP/IP protocols, to be viewed and/or stored to hard disk. The system does not employ any significant image analysis remotely, the distributed processing being limited to video compression and transmission.

In [41], A system is presented where analogue TV signals are captured at remote locations and then digitised and compressed for transmission as MJPEG via an ethernet interface. In this application, the sequences are transmitted using an existing cable TV network to a central location where a high performance computing network independently performs the analysis of the sequences to detect interesting events.

2.4.4 Enhanced remote processing

The distributed processing concept is taken a significant step further in [42] which implements processing locally at the remote camera locations. At each remote surveillance point, the local processing task is subdivided into modules, each implementing a discrete image processing step. The modules implemented in the example system, used for registering abandoned objects in unattended railway stations, are: *Image capture/digitisation*, *change detection* (detection of pixels not due to background which have remained stationary for long periods), *attention focusing* (noise removal using morphological operators), *sub-region extraction*, *classification* (using a neural net to classify the cause) and *transmission*.

Information is transmitted along the network in an alarm instance i.e. where the neural net classifies an event as relating to an abandoned object. In this case the transmitted data is the background image, the extracted sub-image and the coordinates of the detected object. The

transmission system is based on Direct Sequence Code Division Multiple Access (DS/CDMA) techniques allowing simultaneous transmission of the components of the alert bundle. In tests on sequences obtained from two Italian railway stations, a false positive rate of 1.8% and missed detection rate of 3.5% were obtained.

Smart cameras as part of an integrated surveillance system are also used in a limited capacity in [43], employing information fusion with data from traffic flow ground loop sensors and traffic light control systems. In this application, the initial alarm data is sent **to** the camera from the ground loop system and causes the camera to send to storage a five minute interval of scene footage bracketing the incident. The default behaviour in the absence of such an alarm is that the images are recorded over to maximise the efficiency of usage of recording resources.

2.4.5 Network design

The design of systems incorporating such '*smart cameras*' is addressed in [44], where simulations are presented for a variety of patterns of network topology, traffic and interactions. The work assumes an asynchronous transfer mode (ATM) network protocol used to carry constant bit-rate text from cameras in normal situations and high bit-rate video and message data in alarm situations. Control data is sent to the cameras by human operators reacting to alarms by directing PTZ (Pan Tilt Zoom) response and/or manually overriding the alarm.

The work is currently at the stage of providing a foundation for more realistic simulation studies to allow recommendations as to suitable algorithms and network topologies to give maximum service.

The issue of design of real time systems itself has been addressed in papers such as [45], where the importance of composing large systems from small, reusable modules has been stressed. To compose a system with efficient real-time results, the modular arrangement must exhibit *interruptability* and should manage allocation of computation time optimally between the component modules.

From the basis of a modular system design, the components can be implemented using *anytime algorithms* which allow computation time to be traded for decision quality by incorporating progressive refinement techniques (to give interruptability) and knowledge concerning their performance/resource relationship. In this project design in a modular format and qualitative performance appraisal will facilitate ths format as a future development option.

2.5 Background modelling

A background modelling approach which captures a robust and dynamically updated representation of the scene's permanent and semi-permanent features/attributes is a key component of most surveillance systems. Although moving objects can be detected without such a representation, by ongoing differencing between consecutive frames for example, such processes fail if an object pauses or is simply too slow moving.

Most traditional background representation approaches in some way approximate the long term average of values observed at a pixel point. Some weighted average of the values observed, sometimes supplemented by a measure of the variance of these values is still the most common approach.

2.5.1 Extensions of uni-modal background representation

An adaptation of the mean background approximation is presented in [46], which distinguishes between *persistent* and *transient* intensity variations. Persistent variations are characterised by a finite set of intensity values reoccuring over a period at a pixel position (oscillatory leaf motion etc.). Transient variations are characterised by the momentary appearance of intensity values that have not occurred in the recent past at such a point (an example being an moving object point).

The function which is compared to a threshold value, is the output of a motion detection filter, y_n, given by:

$$y_n = \frac{x_n - m_n}{l_n + \epsilon} \tag{2.1}$$

where:

x_n is the current observed value

ϵ is a small scalar constant to avoid possible division by zero

m_n and l_n are estimates temporal mean and difference respectively:

$$m_n = \alpha x_n + (1 - \alpha) m_{n-1} \tag{2.2}$$

$$l_n = \alpha |x_n - m_n| + (1 - \alpha) l_{n-1} \tag{2.3}$$

where:

α is a weighting factor to control the sensitivity of the filter.

l_n acts as a normalisation factor such that, with steady-state or occilliatory variation, the contributions of the variations will tend to cancel out. A transient variation will be lower by a factor of α in the numerator of equation 2.1 (compared with the denominator) and so a response will be given.

An alternative for background representation involves using a Kalman filtering approach as in [47], with the state of the filter at each pixel corresponding to the intensity of the background image at that point. Each new frame's values are used as the measurements for the system, with the updated background generated as the predicted states for each pixel.

2.5.2 Multi-modal background representations

Using a simple average over time is not robust with respect to multiple moving objects and periodic oscillations in a scene (a typical example being a tree's movements in the wind). Simple averaging techniques are also prone to erroneously including shadows of objects and furthermore, slow moving objects can corrupt the background approximation as their values are partially incorporated.

A more complex approach [48] models each image pixel as a mixture of Gaussians. This method is implemented in the current system design and so is described in full in Section 4.4.4. In this approach, individual Gaussians are assigned to correspond to background or foreground at each pixel point based on persistence and variance measures. A pixel is classified dependent on whether the distribution which best represents it is considered to be part of the background model. This type of background can cope well with lighting variation, slow-moving objects, clutter and repetitive motion.

Deficiencies of this approach including a slow initialisation rate and difficulties in distinguishing shadows have recently been addressed in [49] by using different update equations at different phases and use of a shadow detection scheme based on a computational colour space. Improved segmentation results and an increased update speed are possible using these improvements.

Another method using a mixture-of-Gaussians classification model to building an adaptable background is presented in [50]. In this approach, distributions at each point represent either

background, *shadowed background* or *object*. A new pixel value is classified in these terms and used to update the corresponding distribution using an incremental form of the *expectation maximisation* algorithm.

2.5.3 Coping with slow moving and disguised intruders

Special considerations required to handle the case of very remote cameras and individuals moving slowly to escape detection can be addressed by implementing *very slow temporal integration* as employed in [51]. The background model uses a standard mean weighted average update, but the proportion of change due to the current values at every adaptation step is set quite low (e.g. $\frac{1}{32}$ and only carried out every N^{th} frame, with the default set at $N = 64$). Also, for areas with a currently identified foreground object, the update rate is reduced by a factor of four to further mitigate the effects of objects' blending with the background.

This system too uses multiple backgrounds at each pixel, which here explicitly try to model motion-based dislocations. In the current implementation, the secondary background is developed in a supervised batch learning procedure, such that false alarm detection values at a point (with respect to the primary background) are used for initialisation of the secondary background. Where a false alarm occurs in respect of both backgrounds in this phase, the secondary background is replaced by the new value. A third background is built as a copy of an old (5-10 minutes age) image, with objects labelled as such: this *old-image* is not temporally updated and is only used in a cleaning phase (below).

To explain the uses of the multiple backgrounds in this approach, the segmentation step must be considered. As well as a general *global threshold*, employed to remove camera gain noise, a *per-pixel threshold* is used to account for differing spatial variability at pixels (e.g. greater at edge pixels than central object pixels). The sum of the two thresholds constitute a *low threshold* at a point, with a *high threshold* set at 4 times this. To be classed as foreground, a pixel must be above the low threshold and 'connected' to at least one other above the high threshold.

In the context of vision research, 'connected' has a very specific meaning, which is discussed in detail in section 4.5.1

Threshold adaption is applied to each individual pixel. Where a pixel is above the current threshold but, due to the connectivity constraint above, is not classified as foreground its threshold is incremented for subsequent frames. For pixels with a value below current threshold,

this threshold is decremented by a smaller amount for subsequent frames. The net effect of these processes is that, at each pixel position, the threshold will be adjusted to reflect the observed noise variance at that point.

A standard difference image is constructed using these variant techniques applied to the primary and secondary backgrounds and simultaneously a low-resolution difference image (reduced in size by a factor of 4 in each direction) is built. The area of a region is calculated using the standard image, but the connected component algorithm (CCA) used to label objects uniquely is applied only to the low-resolution image, to increase speed. Size thresholding is then applied followed by two processes which look at the intensity for each region normalised using the average of the current image, primary background and the *old-image* background. The first process involves thresholding each region using the normalised background to remove the effects of local lighting variations, the second process involves doing the same using the normalised *old-image* to remove 'ghost image' regions due to dis-occluded background.

This approach is a good example of the improvement of a standard approach using a multitude of enhancements which refine the system's abilities and remove specific limitations. Such enhancements could be applied as modules to improve the performance of a well designed object oriented system.

2.5.4 Coping with fast illumination changes

All background modelling approaches which use dynamic updating approaches to react to gradual illumination changes tend to exhibit problems in situations of relatively fast light changes. Where a cloud passes in front of the sun or artificial lighting is activated, the fast global change can result in a large region of the frame being incorrectly classified as a foreground object.

[52] tackles this problem from a physics-based viewpoint, considering the light from any surface point as characterisable by the surface's illumination and its reflection properties. The work shows that, using an assumption for the chromatic average of ambient objects in a scene, the RGB colour channels can be normalised to separate the variation of illumination from that of surface reflection. Using this normalised RGB colour space, a Gaussian mixture model robust to fast illumination changes is constructed.

2.5.5 Error analysis of background representations

An error analysis of background modelling approaches in isolation is given in [53]. Error properties are analysed in terms of labelling as foreground or background with a *false alarm* being an incorrect labelling as foreground, and a *missed detection* being an incorrect labelling as background. Rather than running a large number of tests, the work uses expectation maximisation and *equilibrium analysis* within the context of a Markov model labeller to estimate the probabilities of the two incorrect labellings for a particular background representation. The probabilities are plotted against each other to constitute a Receiver Operating Characteristic (ROC) graph which can be used to set system parameters.

As part of the results, it was illustrated that, for the scenes considered, a single Gaussian per pixel model was not a good representation but that a static *mixture of gaussians* approach gave significantly better results. Further, problems due to noise were reduced by using a *dynamic mixture of gaussians* model.

2.6 Segmentation

Segmentation, the extraction of moving or otherwise significant objects from a scene, is closely linked to background modelling in those systems which employ such a representation. In many systems, some form of differencing/thresholding is used, either simple (as in [25] [46] discussed above) or complex (as in [51] discussed above). In these, the difference between current and background images is compared with a pre-set value to determine significance. In other approaches (as in [48] [50] discussed above) the segmentation and background representation steps are fully integrated.

2.6.1 Optical flow

An approach which is very useful and common for the case of rigid objects involves the calculation of *optical flow* vectors for a regions. This may be in terms of the observed vector displacements at a point or the estimated velocities: points with similar vector values are directly segmented out as corresponding to a single object. This approach is generally less useful for flexible objects such as pedestrians, whose internal movement can confuse the measure. Recent work [54] has presented a voting method where consistent optical flow data are accumulated

over time to detect humans. This data is integrated with edge, depth and uniform brightness region data using ad hoc heuristic rules to disambiguate between object and background motion and to assist in tracking.

However, the approach to tracking boundaries of non-rigid objects using snakes ([55], [56], [57], which served as the inspiration for the B-spline fitting approach used in [25] has been extended to real-time tracking in combined spatio-velocity space [58]. Using image intensity gradient and *optical flow* as system measurements at the contour, robust tracking in the presence of occlusions and clutter is possible using a robust Kalman filter to detect and reject spurious measurements and occlusion effects [59].

Another method which aims to mitigate the limitations of direct motion segmentation for flexible objects is presented in [60]. Similar to the background approximation in [46], discussed above, this method distinguishes between motion with a coherent direction (e.g. a pedestrian crossing the scene) from that with a random or periodic pattern (extremity oscillation, vegetation in the wind, water specularity etc.). A measure of motion salience is defined which is based upon the extent to which a coherent motion dominates a local area in the spatiotemporal domain.

The input sequence of images is convolved with spatiotemporal gradient filters in both directions in each of two spatial dimensions and the results are squared to give scalar image-like representations of the energy due to coherent motion at each point. Separately for each of the two dimensions, the absolute value of the ratio (*difference:sum*) for the results for the two directions gives two salience maps, one up-down, one left-right. The final overall salience is the result of applying a *max* operation and Gaussian blurring to these two. Thresholding of the image gives a low-noise motion difference result.

2.6.2 Adaptive change detection

An alternative approach to segmentation is presented in [61] which illustrates a method combining aspects of temporal differencing (at its simplest the differencing between consecutive frames to give Δf_i) and background subtraction to give what is termed *adaptive change detection*. Temporal change is detected based on the difference image Δf_i and a weighted accumulation of the last N such difference images. Accumulating the two gives a measure of the changes due to moving objects and lighting changes which is then thresholded to specify

regions of temporal change. This is augmented by background subtraction to isolate objects which are not currently moving but which are also not part of background.

The background is modelled simply, using the mean observed value with the standard deviation used as the threshold value to specify the allowed range. The background is adapted using a weighted accumulation of the mean and variances over the last N frames. To avoid corruption of the estimate by known objects' values, this adaptation is only implemented for the region which has not corresponded to an identified foreground object for a specified number of frames.

2.7 Pedestrian detection

Pedestrian detection is the most basic stand-alone goal of a normal surveillance system. After the previous steps have been carried out, this is the classification of detected objects as 'pedestrian' or 'not-pedestrian'. Classification approaches may be *model-based* where a pre-learned pedestrian model, usually built from an example sequence, is matched to each detected object. Alternatively, it may be model-free (a.k.a. *pixel-based*) where lower level object attributes are used according to some set of approximations and assumptions about common pedestrian characteristics.

The approaches considered, like the system under development, are targeted at remote pedestrian detection. The camera is mounted on a building or pole at a distance from the subject pedestrian area which is relatively high compared to the focal length of the camera. The range of anticipated pedestrian heights in such applications for a VGA format image (640 x 480 pixels) will be in the region of 50 to 300 pixels.

2.7.1 Block-based detection

In this pixel-based approach [62], waiting pedestrians are detected by a pole-mounted camera using frame differencing against an average background image, followed by simply checking that the detected pedestrian does not move for a minimum N frames. To allow processing in real time using a single DSP and to reduce spatial sensitivity to noise, each frame is divided into blocks with subsampling applied to each block. Blocks which do not correspond to a specified waiting area can be 'turned off' manually to further reduce processing requirements.

Detection is performed by thresholding the difference between block averages for the current

image and a mean average background, using an adaptive threshold which is governed by the degree of light at the waiting area. The result is a map of active blocks, a pre-specified sized grouping of which corresponds to a pedestrian object.

Block size, subsampling rate, threshold update factor and number of blocks per pedestrian need to be manually specified during a calibration phase taking several hours, which represents a relatively large investment of time for the simple results offered. Testing the program (written in DSP assembly language) on 23 minutes of video signals at 3-3.5 fps showed that the system consistently over-estimated the presence of waiting pedestrians.

This is an example of a quite simplistic approach to detection, which is only capable of function in a very constrained and pre-specified environment.

2.7.2 Blob features

The use of simple features derived from segmentation is a common pixel-based object classification approach. *Pfinder* [63] uses a more flexible and complex approach, modelling pedestrians using a grouping of 2D gaussian blobs, augmented by a pixel-wise support map. Pedestrian components are represented by individual blob-like objects, specifically using the blob's spatial (x, y) and colour (*YUV*) values, each of the originating pixels having had sufficiently similar such properties to be classed together. *A priori* knowledge about classes (e.g. normalised skin colour) is used to assist in updating a spatial model associated with each blob to predict its distribution in each new frame using a Kalman gain matrix.

The log likelihood of pixels in new frames combined with spatial priors and connectivity constraints allow pixels to be assigned to existing blob classes or initialise new ones as appropriate by comparing the computed class membership likelihoods. This exemplifies the use of a semi-complex model where the distribution and inter-relationship between body components detected as blobs are used as classifier attributes.

Another approach which includes the use of blob features directly in detecting pedestrians is given in [64]. A common background modelling approach is used, taking the pixel-wise mean and variance for each colour channel, with foreground segmentation of the current image by differencing and thresholding at values proportional to the standard deviation. Morphological operations reduce noise and a 4-neighbour CCA is used to label moving objects above threshold size as blobs. The background is not initially updated at areas where a blob is currently located,

but to allow eventual incorporation of stationary objects, if a low velocity is recorded for a specified number of frames, the blob can be explicitly re-classified as background.

The blob features which are extracted are centroid (c_x, c_y), second-order moments (S_{xy}, S_{xx}, S_{yy}) and, for confirmed objects, velocity (u). Second order moments are equivalent to the statistical attributes variance (in the x or y direction) and covariance (in the xy direction). A measure of dissimilarity d between blobs in the five dimensional feature space defined by $(c_x, c_y, S_{xy}, S_{xx}, S_{yy})$ is specified so that matching can be implemented using a 'greedy' search procedure, with matches chosen below a pre-specified threshold on a 'first best' basis.

As blobs can correspond to either a single real object, part of one or a group of them, alternative hypotheses are generated for blob/object matches in local clusters and these are maintained as part of the internal state of the tracker. The most likely hypotheses are output at each frame and evidence from subsequent frames used to update the alternative hypotheses, allowing a revision process which can efficiently cope with objects merging and splitting.

The performance of this approach is very well presented, with all parameter settings listed and results including error rates provided. Good tracking results are shown for a real test data sequence with occlusions.

2.7.3 Complex classifiers

The distinction between pixel-based and model-based approaches is not always straightforward. An approach to pedestrian detection which works on individual images without the need for a background model is presented in [65]. This is a trainable object detection system, which learns how to detect classes of objects in unconstrained still scenes. The object representation uses projections of object images onto a dense *Haar wavelet* [66] basis which encodes structural features at multiple scales. The Haar wavelet transform finds multi-scale edges resulting in a three sets of coefficients (one corresponding to each of vertical, horizontal and diagonal edges) that indicate response over the whole image.

The representative features are chosen as a small subset of N of these coefficients which are consistently strong or weak across the training patterns. The N-dimensional feature vectors are used to train a Support Vector Machine (SVM)[67] classifier for each component of a pedestrian (arms, legs and head). The individual SVMs are used to detect pedestrian components in a new image. The detection is performed on incrementally resized versions of the image to implement

multi-scale detection.

An Adaptive Combination of Classifiers approach is used such that the highest score for each component is used as input to a further linear SVM classifier which gives the final decision as to the object's classification. ROC curves for the approach show that it can detect pedestrians in single frames including occlusion and clutter with a false detection rate of less than one every 796,904 patterns. There was, however, no information available on the time resource required for this level of performance.

This approach illustrates the use of a system wherein the characteristic attributes of a pedestrian are stored as inherent attributes of the model, without recourse to specifying/modelling directly observable attributes.

Pedestrian detection is often the penultimate processing step in a tracking application and more approaches to detection in that context will be examined in the following section.

2.8 Object tracking

Although pedestrian detection is in itself a possible ultimate goal for a surveillance system, most work in the area is focussed on taking the scene analysis at least one step further to implement tracking of such objects over time.

Approaches employ a variety of background representation, segmentation and pedestrian detection techniques followed by methods which relate the results found in one frame to those in the previous one. The most significant step in the preprocessing is the choice of pedestrian detection approach, primarily between *pixel-based* and *model-based* methods, as the results of this step most directly affect the choices for inter-frame tracking.

2.8.1 Flow vector tracking

The tracking problem is generally simpler for fixed shape objects, where approaches based on flow-vector segmentation methods are more suitable. An example of this type of approach suitable for real-time tracking of moving objects (against a background which may exhibit motion relative to the camera) has been developed in [68]. A list of 2D corners is extracted from a frame using the SUSAN corner detector and initial tracking is accomplished by matching against the

previous frame's corner list using a vector description of the feature based on intensity and the *(x,y)* coordinate of the centre of gravity. A Kalman filter is initialised for each feature to maintain tracking.

Individual flow vectors are calculated using displacement over the past N frames and the resulting set of vectors is processed to extract clusters hypothesised as each corresponding to a distinct object. Based on the weighted vector similarity for each of these, the centroid, bounding box and motion model (using a Kalman filter again) are calculated and matched with the previous frame for tracking. Each object has an associated radial shape map which is used in matching and prediction. Occlusion is managed by fixing the shape where overlap occurs, using any observed points to predict shape changes in unseen parts.

2.8.2 Disturbances

The approach taken in [69] dispenses with explicit object segmentation in the tracking of moving bodies entirely. A background is constructed using a standard approach, as the weighted temporal average of recent frames.

The background is linearly subtracted from the current frame to give a difference image, here termed as a *disturbance field*. An analogy to time-lapse photography is useful to help envisage that local motion is characterised by a 'wave-like' disturbance. Each disturbance has a head point which corresponds to the hypothesised current location of a moving object and a sequence of fainter (oppositely signed) tail points which correspond to the object's recent positions.

Tracking particle objects are defined, which contain data of their current position, current state (inactive, active, holding), time spent in current state and, if currently associated with an object, its contrast compared to background and position/velocity history.

For each frame, inactive tracking particles are scattered at evenly spread 'grid points' in the disturbance field and are designed to be attracted to the head of a disturbance if, and only if, they originally lie in its tail. This restriction helps to give good separation for multiple complex trajectories in a scene and can cope with occlusions and collisions by extrapolating the historical path described by the tail. If the head where a tracking particle comes to rest does not yet have an associated tracking particle, the current particle becomes active there and is then associated with that head.

This approach has been shown to be effective in tracking not only pedestrians, but also columns of ants and flowing water. The approach is thus very flexible, due to the absence of complex assumptions as to object attributes and provides an interesting alternative avenue for tracking research generally.

2.8.3 Single blobs

A more common, model-free approach to tracking pedestrians relies on considering 'blobs', obtained by frame differencing and CCA labelling. Although these approaches do not use an explicit learned model of a pedestrian, simple geometric (or other) features of such blobs, which are taken as characteristic of pedestrians are used in detection and tracking, do constitute a kind of simple model.

Such a blob-based approach is presented in [70], which combines low-level image processing methods with mid-level trajectory estimation to deal with occlusion handling. A statistical background model is constructed using the mean and variance values from the image over time, from which moving objects are segmented by computing a distance measure for each pixel in the current frame using a log-likelihood measure. Regions evaluated at above a threshold dissimilarity are marked on a binary interest map and labelled using a CCA algorithm to give segmented blobs.

An Extended Kalman Filter (EKF) is used to predict objects' future positions using assumptions as to trajectory continuity. If a collision is predicted, an *occlusion reasoning* mechanism is brought into action for a potential event. If the relevant blobs then merge, an occlusion event is recorded, the EKF prediction system is used to update anticipated position/velocity and a future blob split is predicted. Division of the merged blob signals the end of an occlusion and the most likely continuing trajectory for an object as predicted by the EKF is assumed.

Another system which uses a single blob representation of a pedestrian and which is particularly suitable for the distributed processing platform domain is described in [71]. The system implements the tracking of pedestrians 'end-to-end' throughout a system comprised of distributed cameras. After frame differencing-based segmentation is used to detect moving blobs, moment invariants are extracted to be the characterising shape features. PCA is used for dimensionality reduction to represent the shape compactly.

Six points on the middle line of the upper body (according to a course 2D human model)

are used for tracking between frames and for specifying three pedestrian features: *(x,y)* position *(location feature)*, the average value of neighbourhood pixels *(intensity feature)* and image:height ratio between consecutive frames *(geometric feature)* are recorded to specify the pedestrian's state. A Bayesian classifier is used to locate the best match between frames using velocity, intensity and position as metrics, requiring that the Mahalanobis distance for the features be below a pre-set threshold to give a match.

To track between cameras, similar features are used, omitting only the geometrical information which does not translate between angles, then projecting the location feature into the same camera coordinates (camera calibration having been carried out as part of an initialisation process). The Bayesian classifier is used for tracking again, using both spatial matching and spatial-temporal matching. Based upon the pre-calibrated camera relationships, an automatic hand-off off the tracking task to the appropriate camera unit is achieved.

[72] specifically considers the case of a large field of view (and so small objects) of routinely crowded areas where segmentation of individual objects may not be possible, rendering both shape and model-based approaches unhelpful. Without such clues, pedestrians are distinguished based purely on *a priori* scene knowledge of their allowed origin points and directions.

The background is modelled from a sequence of 30 frames sampled at 3 fps, with pixels showing a stable intensity over two frames or more used to calculate the variance at that point, updated periodically to adjust for illumination changes.

Initial detection is of all current values which exceed the background variance measure, giving potential object blobs, but this is then supplemented by evaluating the *texture* in the area. For each blob the region is accepted as an object if the texture value too deviates significantly from that of the background. The texture operator is insensitive to incoherent motion and so is intended to eliminate irrelevant signals.

Morphological operators and CCA are applied to generate a list of accepted objects, parametrised by centroid, area, elongation, and mean average grey level. A three step algorithm assuming smoothness of these parameters' frame-wise variations is used for tracking, based on a distance metric in the feature space. This employs a combination of fuzzy logic and Kalman filter update to avoid unreliable associations due to shape variation and occlusion. The first step associates objects moving with uniform predictable motion, the second associates those which do not have such motion, but do have stable parameter values and the third reconnects others

which have had a large deviation in some parameter.

The procedure gave lower false detection rates than the previous blob-based methods, but was unable to track pedestrians well on their merging with other objects. Also, a processing time of 1.7 seconds per frame was noted to be outside the limits of the smooth movement condition.

2.8.4 Blob clusters

A variant approach, already considered for simple detection for [63] above, is to consider certain groups of blobs as comprising single pedestrian objects. Focusing on tracking for an intelligent environment, [73] uses stereo cameras calibrated to the known ground plane of a 'test room'. A segmentation module fuses depth and colour information to achieve a more robust result to shadowing and illumination changes. The resultant maps of foreground blobs are grouped into 'people-shaped' clusters starting with a minimum spanning tree with blob distances defined as the Euclidean distances between blobs' 3D centroids. PCA is used to evaluate the eigenvalues of the covariance matrix of hypothesised blobs to give a measure of shape using the first two eigenvalues (λ_1 and λ_2).

Each hypothesised blob grouping is thresholded for size (using $\lambda_1.\lambda_2$ as the measure) and $\frac{\lambda_1}{\lambda_2}$ is tested against a person-model (which is the expected height/width ratio of a person). The centroids of confirmed persons are projected onto the ground plane for tracking. A coarsely quantised (64-bin) colour histogram is calculated for each person by both cameras to assist in distinguishing between individuals during matches.

The person tracker maintains a list of past locations for each person which is used to compute an (assumed constant) average velocity for predicting the next position. One colour histogram is stored for the tracked person corresponding to their passage through any of 10x10 square cells on the ground plane (which decomposition allows for spatially varying illumination). A local search for current blobs in the area of a prediction is carried out with the best match ascertained based on histogram matching.

The system reports good tracking for up to 3 people in the room with results at 3.5 Hz. The system's usefulness however is restricted to a very constrained and well known environment and a similar approach is only likely to be useful in internal, high risk area surveillance.

A more complex and flexible approach to blob clustering is given in [74], which relies on colour

segmentation of an object into regions in a $2\frac{1}{2}$ dimensional colour space. Based on the observation that same colour objects tend to form clusters in RGB space oriented toward the origin, projection of 3D clusters onto the *chromatic plane* ($Intensity_R = Intensity_G = Intensity_B$) is used to give good unimodal clusters with approximately parallel axes. A subsequent 1D clustering of each 2D cluster can be performed to retrieve any residual colour information, completing the $2\frac{1}{2}$ *clustering*.

CCA is used for unique labelling and a region adjacency graph (RAG) is built to describe the relation of the constituent colour blobs. This is matched against extracted RAGs in areas around the predicted location in subsequent frames to achieve good tracking at about one frame per second. The RAG description is not affected by the shape variation seen in pedestrian objects, and is robust to illumination effects but no occlusion handling procedure is described as yet.

2.8.5 Explicit models

One example of a system which uses a more explicit pedestrian model is the ITPMBVS system described in section 2.3 above and developed in more detail in Chapter 4.

W^4 [75] employs a combination of global shape analysis and local correlation techniques to track robustly with respect to occlusion and pedestrian interactions. The background for each point is estimated using three variables recorded over a specified time window: the minimum intensity $[M(x,y)]$, the maximum intensity $[N(x,y)]$ and the maximum observed change between two consecutive frames$[D(x,y)]$. This background is updated periodically, but only where no foreground object is currently observed.

The segmentation criterion used then is:

$$|I(x,y) - M(x,y)| > D(x,y) \tag{2.4}$$

or:

$$|N(x,y) - I(x,y)| > D(x,y) \tag{2.5}$$

where $I(x,y)$ is the current intensity value of the pixel.

A combination of morphological operations, size thresholding and CCA generates uniquely

labelled objects for each of which centroid, median and bounding box are registered.

In simple cases, the system uses a second order motion model for each object's bounding box to predict its future position, and the predictions are matched with actual object bounding boxes which overlap with a prediction in the next frame. Here, the system updates object position in the motion model using the *median* coordinate as an initial estimate, which is more robust to the effects of extremity movements. Binary edge correlation is performed between current and previous shape's edge profiles to improve long term accuracy.

Simultaneously with this basic tracking, the system creates two sorts of appearance model for each isolated object. The first is a *temporal texture template*, $\Psi^t(x, y)$, given in the paper by:

$$\Psi^t(x, y) = \frac{I(x, y) + w^{t-1}(x, y)\Psi^{t-1}(x, y)}{w^{t-1}(x, y) + 1} \tag{2.6}$$

where:
$I(x, y)$ is the current intensity value of point (x, y) where position is given relative to the median of the region.
$w^t(x, y)$ is the weight at time t, incremented from 0 in line with the number of times that a pixel is detected as part of the foreground.

The second is a *cardboard model* [76] which is used to represent the relative position and size of body components (head, hands, torso, legs etc.). The model assumes people will be in a standing pose and assigns the model height to be the size of the bounding box. Pre-set height divisions are used for head/torso/legs ($\frac{1}{5} : \frac{1}{2} : \frac{1}{2}$ with overlaps) and widths are set using the median width of each component inside it's initial bounding box. The positions of each component are refined using their own temporal texture templates.

In instances where objects merge and separate, the first solution is that temporal texture templates of the original objects are compared with the separated ones to decide the best match. If pose changes sufficiently, this method can fail and a second option is that average intensity values for separate parts of the body can be used. To deal with object splitting, a split is only recognised if it persists for a pre-specified number of frames. Incorporation of a real-time stereo computation into the system [77] has enabled the system to handle relatively fast illumination changes, shadows and more severe occlusions.

2.8.6 Active blobs

An approach which has not as yet been applied to pedestrian tracking, but which offers interesting potential in the area is presented in [78], and models objects for general non-rigid motion tracking using a deformable triangular mesh to capture shape, along with a colour texture map to capture appearance. When the object has been segmented and its boundary ascertained, texture mapping hardware using a modified Delaunay triangular meshing algorithm can be used to build the 2D *active blob* model. Non-rigid deformation of the resulting mesh can be defined using a variety of parametric functions, which will implement deformation of the model.

An RGB colour texture map is captured, Gaussian blurred and mapped directly to the triangulated model to capture the object's appearance. Tracking between frames is now a matter of *active blob registration* to minimise priors on shape and appearance, minimising the squared error for all pixels within the blob. Two tracking examples are presented in the paper, with visually impressive image-like results for regions approximately 2500 pixels in size.

2.9 Behaviour analysis

Whereas pedestrian detection and tracking has been the ultimate level of output for many systems developed over the past decade, more recent research has aimed to take scene analysis a significant step further.

To give the most useful analysis of surveillance footage and to generate the most usefully selective alerts, systems can implement *Image Understanding* approaches to augment the existing *Image Processing* steps. Such systems can take the raw information on pedestrian attributes (tracked trajectories, shape variations etc.) and perform higher level analysis intended to interpret the represented behaviour.

Although the system to be designed will not implement this level of processing and assumes direct output to a human operator, its modular construction (see Chapter 3) allows for future extensions incorporating *Image Understanding* abilities and it is important to consider it in this context.

2.9.1 Learning and classifying trajectories using neural nets

Several approaches implement the automatic learning of normal trajectories in a scene, using a variety of methods. In an alternative approach to Kalman filter based tracking in the ITPMBVS system described above, a neural net can be used to learn the spatio-temporal variations exhibited by the contours of an object [79] and to predict the subsequent values. A compact state vector is constructed using the first five shape vectors and the two position variables. A state vector sequence comprised of the five most recent frames is used to train seven neural networks (one for each parameter) using *back propagation*. The output of the network is a prediction of the subsequent state.

In [80], a neural network is trained off-line on the raw motion vector patterns after background removal and inter-frame differencing to recognise/predict 'usual' motion patterns from a particular camera viewpoint. The system is applied to monitor a scene where motion vectors are used to segment objects on the basis of direction, speed and position. Predictions are compared with actual observations to calculate an error measurement for each moving object in the scene. The error level is a measure of how unusual the motion is and after sufficient confidence is gained (by comparison with an operator-set threshold), an alarm is triggered. Where false alarms occur, operator feedback is used to continue learning by updating network weights to ignore such events in the future.

A neural network, specifically a Self-Organising Feature Map (SOFM) is also used in [81] to learn the characteristics of normal trajectories in a scene and detect novel ones. The approach uses no explicit behaviour models, works in 2D image coordinates and can function on-line for partially complete trajectories. Objects are detected by frame differencing against an adaptive background and labelled using a CCA. The system uses just size and aspect ratio thresholding to eliminate non-pedestrian objects which are then tracked using a simple Kalman filter.

The centroid points corresponding to trajectory are combined with a 'short term memory' of recent position, implemented using a moving average window in a *smoothing* operation. Instantaneous velocity and acceleration measurements based on the smoothed position are calculated and these measures are together taken as a feature vector, $\mathbf{F}$. Such feature vectors were used to train an SOFM using 206 normal trajectories and tested on a set of 23 unusual and 16 normal trajectories. A trajectory was classified as unusual if two or more points were so classified. All of the unusual trajectories were correctly classified, with 19% of the usual trajectories incor-

rectly classified as unusual. This shows a weakness in that the system will incorrectly react to normal behaviour where it is not sufficiently represented in the training set.

2.9.2 Probabilistic models of behaviour

[82] uses probabilistic models (Hidden Markov Models: HMMs) to model spatio-temporal regularities, from ground plane object trajectories. A HMM represents these patterns as a network of temporally dependent belief hypotheses which are updated by adjusting in line with observed visual evidence. The resultant *Visually Augmented Hidden Markov Model* can be used to model and produce dynamic expectations for moving objects and are used in the paper used to direct attention in the image.

In [83], when an object of interest is located, it is assigned an *agent*, which is a piece of modular software often used in Artificial Intelligence (AI). Using AI techniques, the agents perform semantic inference from data, interact by exchanging relevant information and analyse global scene evolution trends.

Objects are identified by position, velocity and acceleration and a B-spline is used to approximate the local trajectory segment. A Baysian network is used to reason over these characteristics to produce the most probable textual description of the instantaneous behaviour. When two objects come within a threshold Euclidean distance, another Baysian network is used to analyse their interaction.

At scene level, HMMs are used to analyse trends of object dynamics. All observed behaviours are combined to build a set of Markov models which constitute a scene history in terms of predefined activities. Although not yet implemented, the models should be able to detect unusual activity when no allowed modelled behaviour provides a plausible explanation of the scene dynamics. Further, such models provide a means of predicting scene dynamics.

Extensive work on interpretation of behaviours using Baysian Belief Networks (BBNs) has been carried out in [84], [85]. The objective of the work is to use the nets to model dependencies between conceptual knowledge relating to a scene and the specific task in hand. The Maximum Likelihood Hypotheses (MLHs) of behaviour from the BBN are used to dynamically vary parameters of low level processing operations to improve segmentation and tracking of objects. The behaviour estimate can also be output explicitly via the medium of state-based activity interpretations in the form of semantic descriptions.

The work is based around a fixed, calibrated camera and requires that a mapping between image coordinates and ground-plane geometry/topology be ascertained in advance. Also, models describing both specified objects and 'interesting' behaviours need to be defined manually. The objects are represented with a generalised cylinder form of volumetric model and cellular decomposition in both space and time is employed to help encode scene information and simplify the behaviour description task.

While the work carried out is of great interest and the integration of the system at multiple levels to allow feedback/control from the BBN is appealing, the requirement for a detailed ground-plane representation is somewhat restrictive and the ability to describe behaviour is quite limited to well known, pre-specified domains.

The ITPMBVS system, [32] [33] noted above, implements a similar approach to pedestrian behaviour analysis. It is combined with work on tracking rigid objects (cars) using a 3D geometric model and then uses BBNs to produce high level descriptions of the behaviour of objects within the scene [86].

2.9.3 Explicit behaviour models

Domain-specific behaviour analysis is possible by explicitly building models of behaviours and classifying observations within this context. [87] builds on the Image Processing output of [24], described above, to perform a domain-specific Image Understanding task. The inputs to this system are the object tracks along with information on object features. The system must be pre-calibrated with *scene context* information including a geometric mapping of image plane to the scene coordinates and a known ground plane. Functional context data is built in to associate labels with scene areas that help to define expected behaviour.

The trajectories are interpreted as collections of primitive events incorporating the contextual knowledge of the scene to derive proximities, speeds and spatial locations. These primitives are classified in terms of a library of pre-specified *scenarios* which correspond to anticipated behaviours. If the matched behaviour represents a forbidden action, an alarm may be generated.

Although interesting work, the paper only gave two image-like representations of trajectory analysis, which did not allow a good evaluation of the system's capabilities.

[88] too uses predefined scenarios, here defined by a state model and the transition between

these states. A configuration phase is again required where the security operator provides information on scene context for each camera. This includes geometrical data specifying interesting areas and their relationships plus semantic information on equipment (type, function, name, characteristics, normal distance and normal time for usage).

Events are defined as spatio-temporal nodes which represent a significant change in the state of objects (person, area or equipment) in the scene. The state of the scene is represented by an n-ary tree containing objects, descriptors, operators and classifiers, with 8 states predefined for the current system. Events correspond to changes between these, with 18 defined in this case, some associated with alarm conditions.

The trajectory of pedestrians is analysed within the known scene in terms of area, proximity to equipment and time spent to define state and event classification.

2.9.4 State machine behaviour models

Another approach to extracting behaviour from a scene is using a state machine model to hypothesise human actions [89]. The system requires prior knowledge of the scene to determine regions (entrances/exits), locations of objects of interest and how an object is used. The low level processes used to give input to the state machine model are colour-based skin detection at entrances, skin tracking (using colour space distance) of people and objects and scene change detection in areas where change is not expected.

A similar self-learning prototype system is presented in [90] where objects are tracked using an extension of the condensation algorithm to give a state vector representation of temporal trajectories. An off-line learning process is used whereby the trajectories are clustered into prototype curves for the scene which can be used to assess all new behaviours observed in terms of position, velocity and time spent stationery.

Combining data on the distances of people and objects gives initial indications as to interactions and changes in this and measures such as *y-position* (for sitting/standing) give the final hypotheses as to behaviour. A log of textual descriptions of recognised actions is constructed and alarms generated if prohibited event/location/time combinations occur. A series of 'key frames' concisely capturing important actions is recorded reducing storage space and transmission times plus facilitating operator analysis.

2.9.5 Modelling body parts

[91] applies a volumetric model of the human body (here using 14 cylinders) with behaviour judged using models of body part relationship variations over time. Once a person is detected, a sub-image is extracted and edge detection applied, followed by line fitting to the edge pixels. A 2D projection of the model is matched with the scene lines, minimising the Mahalanobis distances and then the pose is estimated using a linear constraint minimisation method previously developed for camera calibration. A Kalman filter can give 2D projections over time for each activity which can then be compared with observation to classify the actual behaviour.

[92] uses the pedestrian detection system described in [75] to spot moving people and their body parts. Their activity is specified using a vector of measurements of the body part positions/orientations over a temporal axis. PCA is applied directly to exemplars of specified activities and the models so generated used for activity recognition. The approach requires a separate PCA model for each activity and further for a limited range of viewpoints. The system was shown to perform acceptably in the limited case of differentiating between four variants of walking.

2.9.6 Alternative approaches

As an alternative to modelling trajectories or body part relationships, in [93], after segmenting moving objects, two motion representations are extracted: *Motion Energy Images* which indicate where and how much motion has occurred and *Motion History Images* which indicate how recently (temporally) that motion occurred. The two image-like entities are high dimensional representations of pedestrian movement to which dimensionality reduction is applied. The 7 *Hu Moments* are extracted which give a statistical description upon which PCA is then applied to further reduce dimensionality.

In this work, seven eigenvectors are used to capture 90% of the variance for each movement. Three different classifiers (K-nearest neighbour, Gaussian and Gaussian mixture) were tested for categorising new movement data between eight pre-modelled actions with the *1st-nearest neighbour* classifier found to be optimal here.

The evaluation of pedestrians' behaviours does not always require the explicit detection of individuals as a pre-processing step. In [94], scenes are considered where the behaviour of a crowd as an entity is analysing using several methods. A variant of the DCT algorithm, the

Linear Area Transform can be applied to a crowd image to evaluate the frequency of head oscillations. These stop when a crowd is too dense for free movement and a simple threshold can be used to signal a warning to operators.

A linear Kalman filter can be used to combine measurements based on crowd area and perimeter to give an optimal estimate of crowd numbers to signal unusual gatherings. Optical flow calculations based on block differencing can be used to give a polar plot which clearly illustrates crowd direction and velocity to again signal dangerous variations and also potentially assist in concourse design.

Another approach, based on the use of low-level image features and local statistical models for the recognition of complex patterns is evaluated in [95]. The scene is decomposed into regions where local models of 'normality' are built in terms of the amount of motion appropriate on a region-wise and overall basis.

2.9.7 General event detection

Moving away from considerations entirely specific to pedestrian surveillance, [96] proposes an intermediate representation of video data to assist in general event detection problems. A three-layer algorithm is used to detect events: first motion blobs are detected and colour and texture information are used to extract 76 low-level features for each. Second, a multi-layer perceptron classifies regions based on the extracted features as corresponding to (here) animal, grass, tree, rock or sky.

At the same time *shot summaries* are generated which summarise the current scene context in terms of detected objects, location/size relationships between them and their temporal relationships. Third, a set of heuristic rules about the nature of events in terms of shot summary characteristics is used to infer the nature of events occurring in the sequence.

The approach was tested for detection of a single specified event type on 45 minutes of footage and correctly detected 95% of the instances manually classified.

2.10 Summary

The papers surveyed span camera and system configuration and evaluation, background representation, segmentation, pedestrian detection, tracking and image understanding.

The papers examined, except where noted, follow the generally accepted procedure of illustrating the performance of an approach on a relatively small data set. The results provided are primarily qualitative and often in the form of image-like entities. As there is no common data set, nor agreed standards for specifying one, any quantitative results are not amenable to making comparative analyses of performance.

These drawbacks are principal motivating factors behind the desire to develop a design and evaluation approach which allows some degree of comparative evaluation of approaches.

The evaluation work will, as noted, be within the context of full *system variant* solutions for the overall problem, comprised of modules corresponding to *candidate solutions* selected from the above and from novel variant approaches. The evaluation context will be used in a remote distributed processing unit with limited resources, designed to generate alarms when particular events are detected. The details of this system will be discussed in the following chapter, along with the detailed rationale motivating the choice of candidate solutions.

Chapter 3
Platform specification and system design

3.1 Introduction

The overall aim of the design and characterisation methodology proposed is the recommendation of the optimum practical solution to a vision processing problem from a set of potential solutions. The context in which this methodology is tested is one where there is an absence of quantitative and comparable performance characterisation data available on the algorithms which could form components of the solution. Further, deriving such data for every extant or new procedure individually requires a level of resource input which is not generally available within the context of a single research project nor a commercial development (See Chapter 5).

In summary, the initial problem which is to be solved is the development of a pedestrian detection and location system to run on a distributed processing system with limited local resource availability. The output is to be information sufficient to decide if an alarm should be generated to signal to a remote user that an interesting event has occurred, which will trigger video transmission across the network. The incidence of false alarms is to be minimised to optimise bandwidth usage and operator efficiency. The system is to be used in a semi-automated context and so should focus on human-readable output but should be flexible enough to allow input to additional processing modules.

A primary contention of this thesis is that there is a need for an alternative approach to the evaluation of potential solutions to such a problem and a design approach which promotes such considerations as an intrinsic component from the initial stages. An approach is presented which can be considered in two parts, the choice of appropriate *candidate solutions* for subdivisions of the task (this chapter and Chapter 4) and the subsequent evaluation of a subset of possible combinations of these solutions (Chapters 5 and 6).

This chapter considers the first stage of the process of candidate solution selection, the analysis of the problem and its domain to ascertain the subdivisions to be considered and the criteria for

selecting appropriate approaches. The second stage of the process, the choice of the candidate solutions given this analysis is addressed in Chapter 4.

The analysis task demands knowledge of the platform on which the final instance of the application is to be run, the details of the functionality required, and information as to the environment in which it is to be used. Using this knowledge, an analysis and planning procedure modelled on that used in general object oriented (OO) software design problems will be applied to this domain to construct a modular system suitable for applying a novel performance characterisation approach.

3.2 Intended platform specification

The platform upon which the final version of the application is designed to run is one in the suite of VideoBridge^{TM} platforms developed by IndigoVision beginning in 1994 [97]. The VideoBridge^{TM} units are embedded video processing hardware platforms which, in their most basic operation, perform real-time video compression/decompression and communication over IP (Internet Protocol) networks.

The VideoBridge^{TM} hardware platforms have an on-board DSP (Digital Signal Processor) and allow real-time encoding and decoding of high-quality video at rates which currently extend from 8 Kbps(kilobits per second) to 2 Mbps(megabits per second), dependent on the particular release and upon user specification. As well as the VideoBridge^{TM} hardware, which can be in the form of boxed or PC-integral codecs the platforms are available as pre-compiled software objects which can be programmed into Flash memory or linked with user-interface software for embedded applications.

Whichever the format of the VideoBridge^{TM} platform, the basic functionality is similar and can be illustrated using the example of the VP400 board shown in figure 3.1. In this release the central processor, the *video processing unit (VPU)*, was a 32-bit reduced instruction set computer (RISC) with specific extensions for processing video data. The processor runs the proprietary *CamOS* operating system and its applications which handle routines including task scheduling, memory and device management, timing and network interface.

CamOS applications are embedded i.e. compiled and linked with the operating system to give a single executable. The applications can be run from non-volatile flash memory (access time

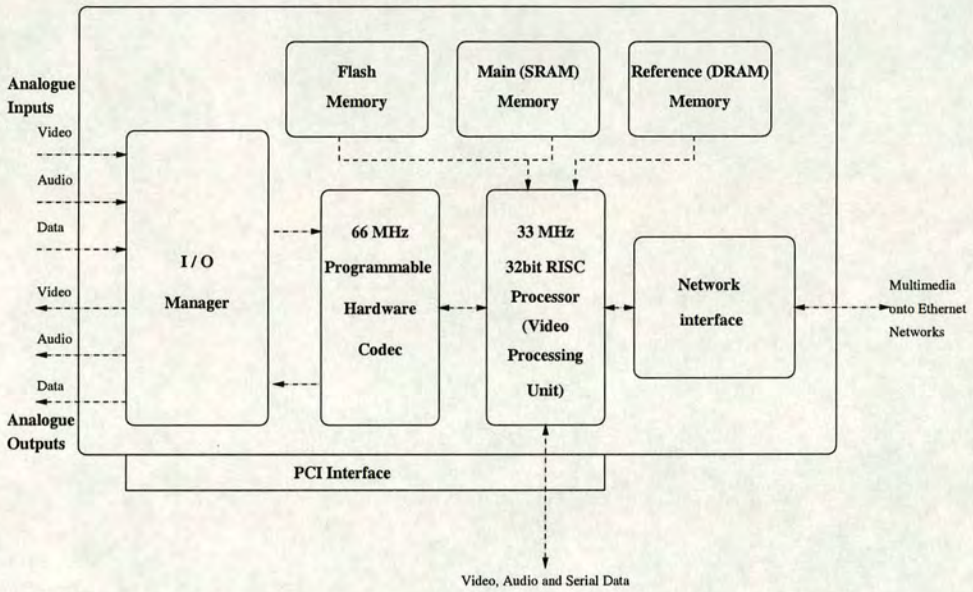

Figure 3.1: *An overview of the VideoBridge^{TM} VP400 board*

time 120 nanoseconds) or from main SRAM (static random access memory), which is faster (access time 15 nanoseconds) but takes up memory which would otherwise be available during the running of the applications.

In the VP400, the memory specifications were:

- 1M x 32-bit SRAM
- 1M x 32-bit Flash
- 512k 32-bit fast page mode DRAM

Captured frames and intermediate processed images are generally stored in the two dimensional DRAM (dynamic random access memory) reference memory, which supports access times of about 40 nanoseconds.

The functionality which the VideoBridge^{TM} offers is, at its most basic level, that an interface is provided through which digital and analogue cameras can be interfaced with an IP network at multiple points on one or many LANs. Data can be sent along the network and received at PCs or analogue monitors on any LAN connected via the internet globally or via wireless transmission, on a mobile device. An example of a networked CCTV application using VideoBridge^{TM}

is given in figure 3.2.

Up to six cameras are connected to the network using each unit of VideoBridge^{TM} hardware in one of its forms, configured to be a *video server* for the system. The video input from the camera can be YUV 4:2:2 digital video or NTSC/PAL analogue (which is digitised by the *video server* as an initial step). The video is compressed, typically in the ratio 80 : 1, applying a user-defined compression standard selected from options including Motion JPEG, H.261, G.711 and G.728.

The compressed data is prepared in an ethernet packet for transmission along the ethernet network using TCP/IP network protocols. This allows a simple 100 Mbps ethernet network to carry more than 200 such video/audio streams (over 2000 if an internet switch is used). The video server can also transmit serial control data (RS232/485) for features such as PTZ camera control and 16 TTL alarm data. The use of TCP/IP protocols for packetising and transmitting data is a key feature of the platform, allowing use on all internet capable systems and transmission of data to all such systems at any remove globally.

One or more PCs are used for real-time administration and control and can also be used for viewing the output using windows-based software applications. Viewing is via a VideoBridge^{TM} codec, configured to be a *video client*, which implements the decoding and decompression required to allow the footage to be viewed via PC software and/or analogue monitors. The image resolution is user defined (CIF, SIF or QCIF) allowing a dynamic balance to be achieved between resolution and band width usage. Using an additional software add-on, a PC can be used to record up to 50 video and audio streams at up to 30 frames per second with simultaneous playback during recording available. The recording can be continuous or may be triggered by events in the scene via transmitted alarm signals.

Using digital CCTV connected by video-over-IP in this way instead of the traditional analogue systems with dedicated wiring can give reductions in capital investment of 70% and in operating costs of 90% [98]. Furthermore, the scenes can be viewed anywhere in the world and with the latest developments can be delivered over wireless networks to hand-held units. This latter advance would allow security guards to view camera output while mobile in a surveillance location, rather than being tied to a central monitoring suite.

The CCTV system can be enhanced by applying additional processing at the *video servers* so that, for example, video is only transmitted along the network when some event of interest is

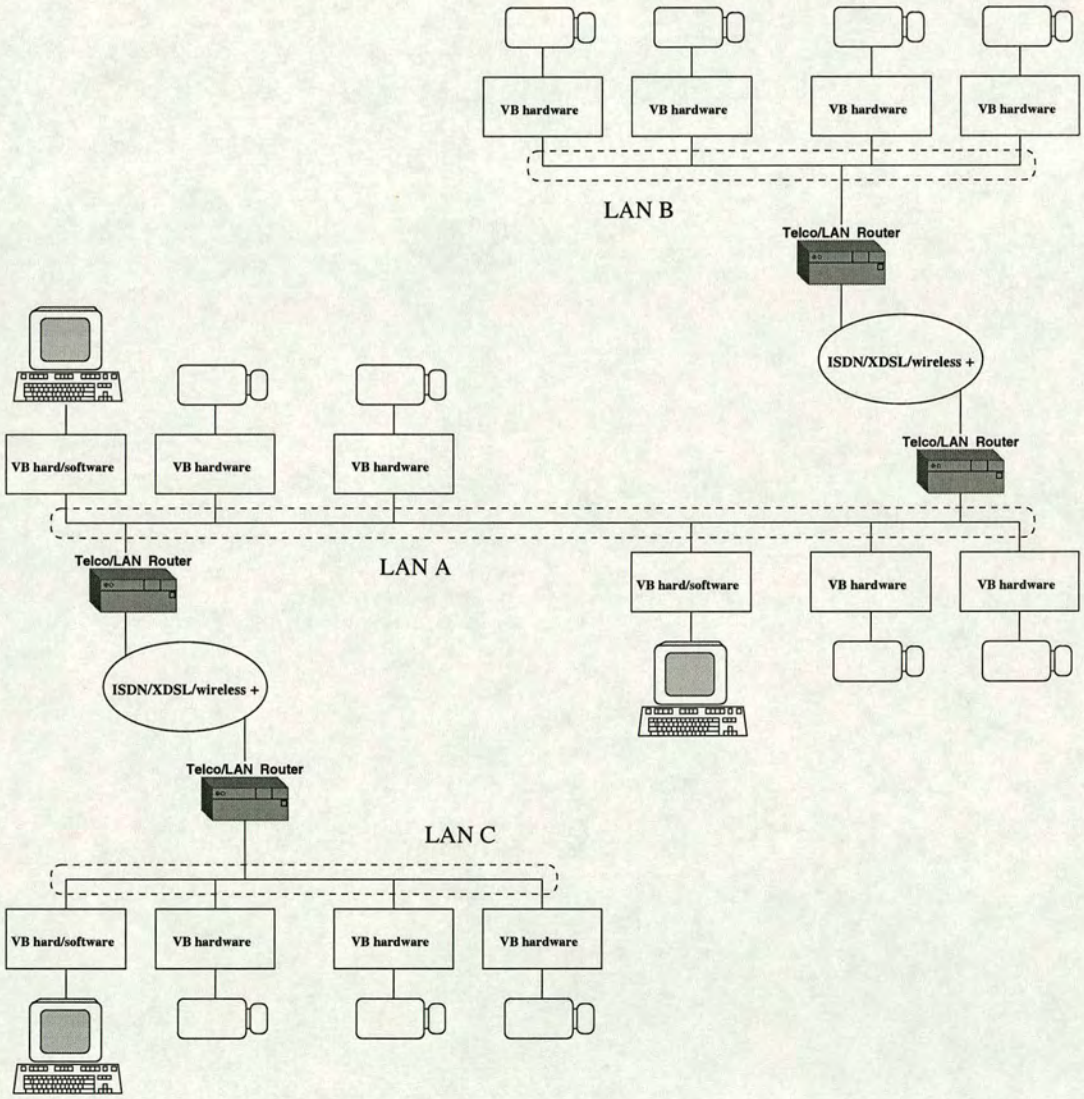

Figure 3.2: *A networked CCTV example using VideoBridge^{TM} technology*

detected in the incoming video stream. At the outset of the project, a simple motion detection system was used in this respect to assess the stream content. In this case, a *video client* signals to a *motion detection server* (a special case of a *video server* set up to perform the detection task instead of constant video generation) that it is interested in any motion detected at one or more cameras. When the server detects motion, a message is sent to the client which makes a decision as to whether to request video.

The benefit of this is that video is not transmitted when there is nothing of interest in the scene, which saves bandwidth usage in the network. As the network is a shared resource with the entity's other linked applications, this is is a significant factor. Also, where there is a human monitor, their attention can be improved by eliminating irrelevant data (see section 3.4).

The use of motion detection as the measure of interest in a scene is the most basic level of processing which is useful in this respect. It was a particularly appropriate development for the VideoBridge^{TM} system as it was able to reuse some of the processing abilities already provided as part of the compression system. The motion detection used a frequency-based analysis where the discrete cosine transform (DCT) is applied to compare between 8x8 blocks in the the current and previous frames. If blockwise significant differences indicate a moving object of suitable size and shape, a motion detection event is signalled [99] [100].

A significant goal of the current project (see section 1.3) was to design a suitable application for registering interest in this manner to be used on a future release of the VideoBridge^{TM} series. The two key general criteria are that a more complex analysis of the current level of interest at a camera is desired and that the resources required to achieve this are constrained within those available at a remote *video server*. This resource consideration must include an estimate of the detection rate given the computing resources as an integral evaluative element. A closely associated requirement is that the level of actual video transmission along the system should be kept to a minimum which means false alarms must be considered to be very costly in the evaluation.

Apart from the general challenges inherent in this task and as detailed in the following sections, there was an additional complicating factor in this specific instance. The rate of development of the VideoBridge^{TM} system is particularly high and at the outset, no confident estimate was available to specify resource requirements which would be available for the final application. As the current research task was conducted apart from the overall development process, a more

general criterion for estimating resource costs was chosen.

On the basis that it was desirable that the *video server* be able to run additional processing tasks simultaneously if possible (task-based multitasking being implemented in *CamOS*), the resource requirement was treated as a minimisation problem. A quantitative analysis of the resources demanded would be performed on the test platform to allow comparative review of the options and a qualitative cost/benefit review conducted. The final evaluation of the application after the main testing was to be on the final platform itself (but see section 3.8).

3.3 Application profile

3.3.1 User context

To complete the general statement of the problem, consideration needs to be given to the context in which the system is likely to be used and thus what will constitute an interesting event. The application area considered, CCTV, is primarily used for a variety of surveillance purposes including the general monitoring public places, people counting, single-camera home security, retail security, secure establishment surveillance networks and speed cameras.

In choosing a particular focus for the task, it was desired to find an application which offered both significant real-world usefulness and potential that individual results could be of research interest in and of themselves.

A prima facie goal of any application to run distributed at the individual *video servers*, as stated above, is that transmission along the system should be kept to a minimum. It is assumed that after a real interesting event is detected, there will be such a transmission to the *video client*, where additional processing of some kind will occur. This processing may be a further automated procedure or a manual (i.e. human) review of the alarm. Whichever is the case, this high level processing can initially be considered to be separate from the distributed processing carried out at the server.

Chapters 1 and 2 introduce the range of surveillance approaches from the traditional all-manual version through to fully automated systems utilising complex high-level behaviour analysis methods. Although interesting in a pure research context, fully automated surveillance outside the realm of science fiction always at some point refers a decision to a human operator. Even if the central processing were carried out automatically within the entity, a final action will be to

alert the police, security or possibly the fire service.

Where, as in this case, the goal is to implement a distributed processing solution to give an initial alarm, the area where this factor is most significant is in considering the use to which the results will be put and the nature of the user. If high level automated processing is used, it can be considered as an additional 'layer' between the system under consideration and the final, human, user. In current commercial applications, the high level processor used is almost universally the human brain, constituting a semi-automated surveillance approach.

For our design purposes, it is most reasonable to assume the existence of a central human user, as this bears out in current practice, there is no general format required by diverse high level processing alternatives and, as noted, the **final** user is always human. The human user may be monitoring the system in real-time and/or may require stored footage along with the relevant alarm/event information to review off-line

To briefly recap the earlier points made in section 1.1.3 in respect of purely human surveillance, two major difficulties are information overload (in situations where all scene analysis is the central user's responsibility) and reduction in attentiveness over time (which is exacerbated by a continual, intense monitoring load). If the information is pre-filtered, initially to save transmission bandwidth, then the information that is received should be sparser and more relevant. The attentiveness of a human operator should be maintained at a considerably higher level. Thus the distributed processing paradigm which applies to the VideoBridge^{TM} system is very appropriate for the construction of effective semi-automated surveillance systems.

3.3.2 Pedestrian detection

The majority of the surveillance applications noted in section 3.3.1 focus on moving human objects, pedestrians and their behaviour in a scene. The task then, will be to develop an approach for generating alarms based on some aspect of pedestrian behaviour, which will then allow on-line human review or storage of image sequences and alarm/event information.

The range of options for analysis of pedestrians in surveillance scenes is extensive (see Chapter 2) and ranges from basic pedestrian detection, through tracking and trajectory analysis to individual identification, Baysian approaches to behaviour analysis and spatio-temporal modelling.

As the problem domain has been specified as semi-automated surveillance, consideration of

possible analysis tasks will be limited to exclude high level processing approaches. These generally duplicate or approximate the high level processing abilities already on hand in the human observers. Also, such procedures would probably need to be implemented at the central processing point rather than the resource-scarce distributed level and so are outwith the scope of the current problem.

Again drawing on the knowledge that the high level processing ability of the human user is on hand, and that some stimulus in the form of possible alerts is likely to increase attentiveness, it is possible to focus on the simplest approaches. The initial goal of any such system is the detection of a pedestrian in the scene, and the estimation of their position. The simplest extension of this (and so the one which uses minimum resources) is to use this result in combination with the existing functionality in the motion detection application which generates alarms when motion is detected in a pre-specified image area.

3.3.3 Application scope and extensibility

This enhancement of the processing would allow a greatly improved selectivity in terms of the events which cause alarms. For a signal to be sent along the network, an object must be segmented from a background representation, classified as a pedestrian and located within the specified area. If the human monitors are signalled only in such circumstances, and further can be signalled remotely rather than at a central location, they can potentially undertake additional tasks simultaneously, giving a more efficient use of manpower.

While this level of functionality is appropriate for the initial investigations, it would be prudent to consider ease and range of extensibility as an important attribute. This is important in two specific areas: first, the processing power available at the distributed processing sites will continue to be increased, allowing greater flexibility for the processing performed there. While wise to be prudent in the initial development, the ability to enhance functionality in line with resources would be very valuable. Such extensions could include tracking, individual identification (by capturing face images or gait characteristics) and additional behaviour analysis (using trajectory modelling or outline spatiotemporal models).

Secondly the system should allow for the previously noted possibility of additional processing at the central location, as an intermediate step prior to notification of the human observer. In this situation, there would be a two-level alarm procedure: the first would be between the *video*

server and the *video client* as described, signalling interest in the scene in question. A level of additional processing using the central resources could then be brought on-line on either the live video stream or the stored data which caused the original alarm. The result of this second level of processing would be a decision as to whether or not to generate a second stage alarm to alert the human operatives.

This additional processing would be higher level analysis of the scene, perhaps using information fusion, spatiotemporal models, face recognition or Baysian statistics to generate more complex hypotheses as to the likely nature of the scene events. Such processing would move the overall approach closer to a fully automated status, although still relying on the human for the final processing step.

It is important then to design a system such that the output is not exclusively geared to direct human interpretation, but which is amenable to future modification to providing input to further modules within the *CamOS* framework.

3.4 Problem specifications

Now that the general problem has been defined, it is necessary to methodically summarise and to provide additional details on the points stated and define some further specific parameters.

Scene specificity In the anticipated CCTV context for the application overall, a complex surveillance task over multiple scenes/viewpoints is envisioned (see figure 3.2). The central processing station (here one or more human operators) must be capable of dealing with data from all of these scenes simultaneously. The application under design, however, is intended to run at one of the distributed processing nodes and so can be configured to process the data from a single scene in isolation. Any decisions based on data from other scenes/sources can be assumed to be handled centrally.

Camera parameters Cameras used in the system may be fixed, stationary units or may have PTZ (pan, tilt, zoom) capabilities. Within the CCTV context, camera translation is less likely, although still a possibility. If the problem is configured to allow for automated PTZ (or translation) scanning, the problem becomes considerably more complex. The platform's existing motion detection procedures assume fixed, stationary units in the specification of regions of interest in the image plane and the cameras in most applications

Resolution	**size** (pixels)
CIF	352×288
SIF	352×240
QCIF	176×144

Table 3.1: *Resolutions available from VideoBridge^{TM} codecs*

are hypothesised as being such units.

Accordingly, the problem is configured to operate on such fixed, stationary units: assumptions will be made during algorithm design which rely on this specification. In the VideoBridge^{TM} system, *video servers* for PTZ cameras can accept control input from the central location to redefine the cameras internal parameters. In these instances, where dynamic PTZ behaviour **is** occurring, the event detection system will not function correctly. It is a reasonable assumption, though, that where the PTZ parameters are being dynamically varied, it will be because monitoring is in progress by a higher level processing unit (here, a human operator) and so the loss of signalling data should not be a problem.

When the high-level intervention ceases and the PTZ characteristics become stable, the event detection approach will begin to function correctly again at the restored or new settings. It is important to note at the outset that many approaches will exhibit a latency (equivalent to system re-initialisation at new settings) before the output results can be considered to be reliable.

Input The input data to the system will be a sequence of uncompressed digital images. The system should be able to handle single-channel (monochrome) and multi channel (colour) images and process each appropriately. The *CamOS* routines which capture a frame can accept direct digital input or convert analogue NTSC/PAL input to a video image, with user specified resolution. The resolutions supported are scaled from NTSC/PAL by the hardware and are listed in table 3.1 and are specified by the user at the beginning of a capture session. Thus the system must be capable of accepting and processing different image frame sizes but need not be able to adjust to the size varying dynamically in a single run.

Output The output of the system should be information sufficient to make a decision concerning whether an alarm is to be sent to signal an interesting event. The procedure for the alarming itself was already available from the *CamOS* routine used in the motion de-

tection server. To allow for easy extensibility, the information available should, where possible be maximised subject to acceptable resource demands. As object oriented (OO) principles are to be employed, the chosen option is to develop towards creating an 'event' object which contains all the information required to make the alarm decision. These objects will then be used in making the alarm decision at each frame.

False alarm rate The alarm signals are to be transmitted along a computer network which may be used in common by many related and unrelated applications. Although the initial alarm signal will occupy negligible bandwidth, the anticipated reaction will be the transmission of a live video stream along the network for immediate analysis and/or long term storage. It is vital then, from the point of view of bandwidth utilisation, to minimise the number of false alarms generated by the system. When selecting candidate solutions and evaluating performance, this requirement must be considered.

Human monitoring The application context is semi-automated surveillance, i.e. the alarm and subsequent data will be analysed by a human operator. The system is essentially implementing a filter on scene information to select that which will be of most interest to the operator. As such, performance can be judged in comparison with a human's image analysis capabilities, with the best performance taken as an individual monitoring a single image for a short length of time.

The efficiency of human vision analysis reduces where there are multiple scenes to attend and where attention must be maintained over a long period. The ground truth derived using human consideration should thus ideally be built in stages from review of single scenes for short periods containing numerous interesting events.

System extent The system will first be developed to detect pedestrians in a scene and specify their position. A simple tracking facility will be included as this assists in detection in later scenes. The system should be open to extension, with interest in face extraction and trajectory analysis as significant areas for development in the short term.

Environmental robustness The application may be run in a wide range of environments. An indoor scene with constant lighting is likely to be the simplest scene encountered, but the application may be required to run externally in an environment subject to a wide range of variation. The minimum variation externally will be the diurnal lighting changes and these are likely to be supplemented with meteorological changes such as rain and/or snow fall, rapid light changes due to occlusion of the sun by clouds, background movement

due to wind and rapid changes due to artificial lighting. The system must be sufficiently robust to cope with such changes.

Resource restrictions The resources available in terms of processing power and memory to be available at each distributed node are not quantitatively specified. The development rate of the VideoBridge^{TM} technology is very high and brings discontinuous increments in the resource availability with each release.

Irrespective of the absolute level of processing which will be available, however, it is qualitatively specified that resources will be scarce. Other applications could be run at the node using *CamOS*' task management protocols and so the minimum resource utilisation is to be aspired to. It is sufficient then to attempt to minimise resource usage in the pre-selection and construction process and to present a comparative analysis of the resource requirements of potential solutions.

3.5 Research goal

3.5.1 Goal summary

As indicated in section 1.3, a core aim of this research endeavour, is to create **and evaluate** a methodology for the analysis of a vision processing problem to run according to user specifications and the design and characterisation of an appropriate practical solution to that problem. As part of this procedure, novel system components and novel extensions/adaptations of extant approaches will be constructed and their evaluation in this context will provide further results of research interest.

The methodology will be tested using the VideoBridge^{TM} design example as a pilot problem, but is intended for general application to vision processing tasks generally.

3.5.2 Current development and evaluation paradigms

As developed in full in Chapter 5, the ideal starting point for the development of a solution to a vision processing problem would be where the range of vision process algorithms presented in the literature had full quantitative performance characterisation data available. In this ideal situation, there would be information on the performance on standard test data sets, the ranges over which it operates, the failure modes of the algorithm and its resource requirements on

common platforms/processors. These results could be put into arbitrary combinatorial chains, using error propagation to ascertain the predicted performance and resource requirements of the composite system.

This is not the situation which a current developer is faced with: most results available on previous work are relatively qualitative and with little to allow comparability between them or prediction of performance in a novel setting. Performing such algorithm-wise performance characterisation on new or re-implemented solutions would require a vast investment of resources (see Chapter 5) and can be safely assumed to be outside the budget both of most commercial design programs and individual academic research projects.

The standard approach in an industrial research and development context starts with system specification by a user/customer. The requirements are translated into an initial systems design by the engineer and validated with the user to ensure that the requirements are adequately fulfilled.

The system specification will include details of the operating conditions, sensor type, accuracy requirements (in terms of detection and false alarm rates) and resource levels (in terms of speed of operation and computational/memory resources available). There is currently no systematic approach for translating the specification into a detailed design. A common 'ad hoc' approach is to assess the qualitative results presented in the literature to make an evaluation as to the algorithms which will best combine to give the desired results on the system. The best theoretical combination is then implemented and the performance of the application overall is evaluated against the desired system attributes.

The choice of the overall system architecture and the specific modules to implement to accomplish the task relies on the engineers' experience to achieve an acceptable system. A systematic engineering methodology for the design and subsequent validation has been presented in [101] which relies on algorithm level performance characterisation and error propagation. Until detailed algorithmic performance characterisation data becomes commonly available, incrementally over time, a compromise approach is needed between the rigorous ideal and the single monolithic implementation alternative.

It is worth noting that the design problem becomes more complex in respect of distributed processing systems, as noted in [102], consideration must be given to the allocation of processing between the distributed units and the central 'hub' processor, scheduling of tasks, bandwidth

constraints and subsystem independence/cooperation issues.

3.5.3 A novel development and evaluation paradigm

Again, detail of the suggested approach is given in Chapter 5, but the general procedure follows the principles of objected oriented modular design, albeit at a slightly higher level. As in OO design, the first step is the analysis of the overall problem into n components but here the components correspond to sub-problems which have been the subject of previous research endeavour in the vision community. A review is then conducted of extant approaches and solutions to the sub-problems available in the literature. Their results are evaluated in the context of the current problem domain and knowledge of the system specifications. This is essentially a formalisation of the first step of the current ad hoc approach, applying the results of the analyses conducted in sections 3.2 and 3.4 above.

Based on both this analysis and an evaluation of any novel approaches conceived for solving sub-problems, m optimal methods for each of the n components are selected as *candidate solutions*. Together, these allow the specification of $(n \times m)$ *system variants* which can be implemented to solve the overall problem. The decision as to the value of m used will involve careful consideration of the resources required to implement the candidate solutions, balanced against the importance of performance optimisation in each project.

A key assumption here is that the module performances are **not** independent of each other within a planned composite system. This assumption is supported by the work on algorithm-level performance characterisation of vision processes as presented in section 2.2. In this work ([18] [19] [103]), the effects of error propagation due to the results from early modules being used as input for later ones are detailed. The overall performance and failure modes of the system cannot be extrapolated from the result profiles of individual components, without extensive and complex statistical modelling.

There will be a total of p proposed modules:

$$p = \sum_{i=1}^{n} m_i \tag{3.1}$$

where m_i is the number of *candidate solutions* selected for module i. Each of the p modules is implemented using OO design principles as a self-contained unit with interfaces designed to

allow subsequent combination into an overall system. Each module is tested independently to ascertain correct functionality as described in prior work being re-implemented/developed or as projected in the design process for novel approaches. The set of possible combinations of theses modules constitutes the *system variants* for the problem.

After the modules are individually validated on small test data sets, each system variant is rigorously tested on a large real data set including examples of all situations over which the final system is to perform. The results must be quantitative in nature and suitable for making a comparative analysis of the suitability of the system variants to solving the specified problem.

Both the results and the test data should be made available to third parties to assist in solving similar problems, providing a quantitative performance characterisation specific to the current problem but from which more general results may be extrapolated. The greater the volume of data used and the range of environments represented within it, the more general applicability the results are likely to have.

3.6 Design considerations

The first step of the evaluation process is the analysis of the overall vision task to be performed into components suitable for implementation as modules. As noted, the process closely mirrors that used in general object oriented design, sharing as it does several of its key goals. In OO design, the aim is to build software in component form, in a manner that has been likened to the use of standardised primitive components to construct complex and varied systems in electrical engineering. Accordingly the desire is to construct reusable components which serve specific and clearly specified purposes and which can be built and tested individually prior to use in a range of composite systems. It should be possible to interchange corresponding component options in a system without redesign as long as they share common interfaces with neighbouring components. Key attributes of good object oriented design can be summarised as follows:

Modularity: Logically related data and the functions to act upon it should be collected into a discrete unit with a stable interface with other such units. These units correspond to *classes* in C++, a specific instance of any class being an *object* of that particular type. These objects are the basic building blocks from which OO applications are constructed.

Functional abstraction: Units can perform the required processing on their input to give res-

ults in a specified format without the details of the processing being known. This *information hiding* implements the requirement that modules performing the same overall operations using contrasting algorithmic approaches may be interchanged within the body of the system without the need for redesign. It is important that such comparable modules be designed with a common interface structure to facilitate this option.

Encapsulation and data abstraction: The data structure of an object and the operations which are allowed upon it are defined together within the appropriate class structure. This implements protection against inappropriate operations being applied to objects which may give undefined results and errors.

Hierarchy and inheritance: Objects which hold significant properties in common should be grouped together. A class is defined for objects which posses only these common attributes and which is referred to as the *parent class*. A group of secondary classes can then be defined as *subclasses* of this one, each of which extends the core object definition in different 'directions' by adding attributes for each subclass which are not shared in common. An object *inherits* the properties of the parent class and supplements these with those of a specified subclass to give an entity with all requisite attributes.

Polymorphism Where suitable, operations should be defined to function on objects of a parent class: when an object of one of the subclasses is passed to the operation *dynamic binding* (i.e. at the program's run time) should select the appropriate processing options for the specific subclass.

These principles will be applied in the 'bottom-up' design process at the individual object level (section 3.8). The benefits of such an approach include the relative ease with which 'off-the-shelf' solutions to individual processing problems can be integrated with in-house developed solutions where they share an OO design structure.

At the *candidate solutions* module level a subset of the OO design considerations will be applied. Each such module will be comprised of one or more OO objects together constituting the identified vision process. *Modularity* is implicit in this structure and *functional abstraction* is important to achieve mutual compatibility between candidate solutions for successive modules. *Encapsulation/data abstraction* is implemented in as far as it is in the constituent OO objects. *Hierarchy/inheritance* and *polymorphism* are less significant at this level but could be used to facilitate module interchange by defining the module solution for each component in terms of

a parent superclass. The alternative solutions would be subclasses of this, which could save development resource in as far as the alternatives share common attributes.

3.7 Top-down design

As noted, the specific choices of module definition for the current system are motivated by its decomposition into sub-processes corresponding to recognised discrete vision problems. It is in these areas that previous research efforts have been focused, leading to a proliferation of approaches for which qualitative performance results may be found in the literature (Chapter 2).

A review of the literature in each discrete area provides the initial criterion using which candidate solutions for implementing each module can be pre-selected. The pre-selection process is considered in detail in section 4.2, but in summary includes available performance results in similar applications and estimated resource requirements.

For the current system, these general considerations are supplemented by an evaluation of how interesting from a pure research perspective certain solutions would be. This could be by virtue of the creation of a novel version of an extant approach which has been extensively modified to work on the final platform. It could also be that a quantitative comparison between certain alternative methods to a solving particular problem has not been previously recorded and so would be of interest.

The stated vision problem then is the development of a pedestrian detection and location system. This can be decomposed into the key modules as illustrated in figure 3.3.

1. **Image capture and preprocessing**: Conversion of input digital video into frame sequences suitable for processing. This is carried out by extant *CamOS* applications in the VideoBridge^{TM} system, but must be implemented on the test platform as the initial module.

2. **Background estimation**: Construction of a representation of the underlying scene background. This should be capable of dynamic variation over time as the scene environment parameters change.

3. **Foreground segmentation**: The extraction of significant objects in the current frame

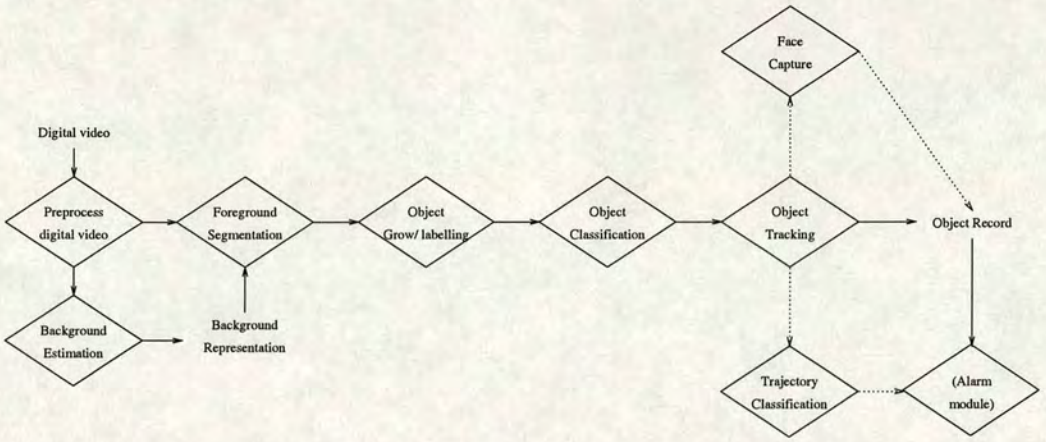

Figure 3.3: *Component modules comprising the vision system*

which do not correspond to the scene background.

4. **Object growing and labelling**: Processing the segmented foreground representation to uniquely label discrete objects composed of connected points.

5. **Object classification**: Considering each uniquely labelled object to classify it as being pedestrian or non-pedestrian.

6. **Object tracking**: Updating a pedestrian object's position and predicting its position in the next frame.

7. **(Trajectory classification)**: An optional extension module: analysing a pedestrian object's trajectory to evaluate its 'normalcy' and make predictions.

8. **(Face capture)**: An optional extension module: capturing a sub-image containing a pedestrian object's face to store for subsequent identification.

9. **Alarming**: Using data from pedestrian objects (which correspond to the *event objects* discussed in section 3.4) to initiate an alarm. This is carried out by an extant *CamOS* routine on the final platform and is not implemented on the test platform.

As detailed in section 3.6, the system should be implemented so that, as far as possible, the selection of each module candidate solution may be made independently. However, the control flow of the latter part of the system is unavoidably affected by the choices made in respect of the object classification approach (see section 4.6). The alternatives can be broadly classified as

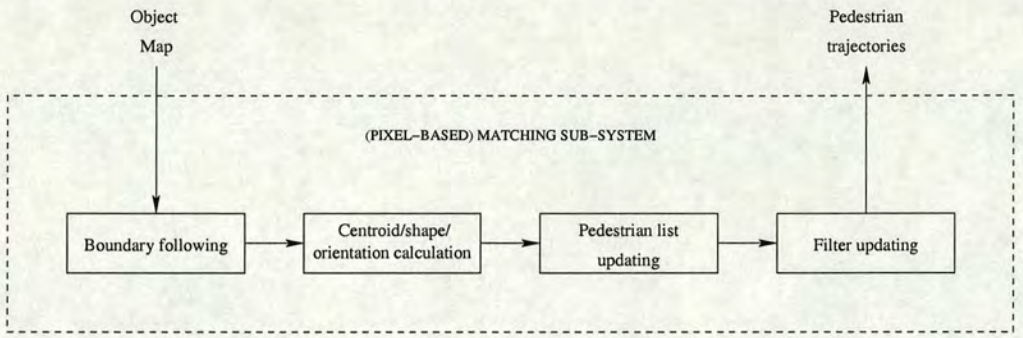

Figure 3.4: *Pixel based object classification and tracking*

model-based and pixel-based methods: the control flow variations after that point are illustrated in figures 3.4 and 3.5.

3.8 Bottom-up design

After the specification of the key modules in the top-down phase of the design process, it is necessary to go below the level of these modules, defined as a key part of the current evaluation process, to consider the OO design specified objects themselves.

The context for this process is the *Vision Systems Code Library* of the *Department of Electronics and Electrical Engineering* at the *University of Edinburgh*. This a repository of code and utilities shared between members of the *Vision Systems Activity* of the *Integrated Systems Group* of that department. To enable effective code sharing, a uniform coding style has been adopted and certain commonly used classes are fixed for use as primitive elements of new applications. All classes of objects designed during the development of the system are written in C++ because of the benefits its encapsulation offer for shared code projects.

The classes *VS_frame*, *VS_frame_io*, *VS_image_process* , *VS_kalman* and *VS_matrix* discussed below were pre-existing classes within the *Vision Systems Code Library*, primarily written by A.M. Peacock and C. Haworth. The new classes written for this project (table 3.3) were designed to interface with and/or derive from these base classes.

The *VS_frame* class is used to store an image as a 2D rectangular array of pixel values. Pixels are stored as integer (int) values which can be displayed directly and allow the use of fast fixed point arithmetic. These can be one or more channels of 2D pixel arrays, the channels comprising a

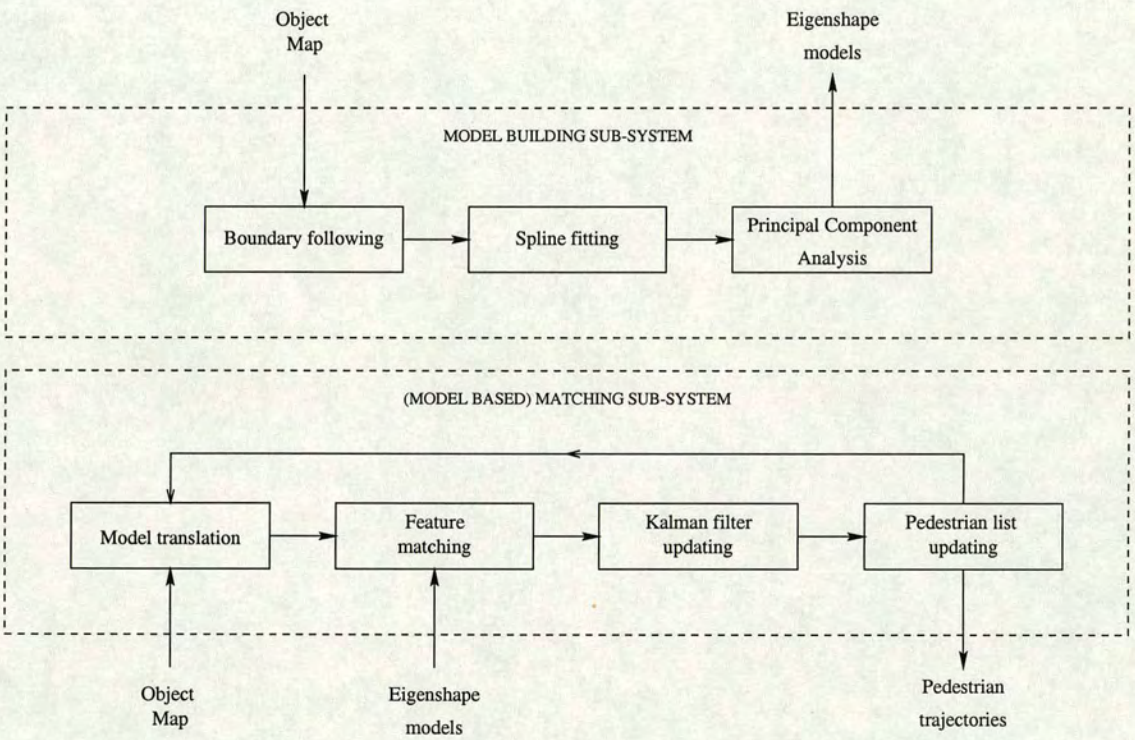

Figure 3.5: *Model based object classification and tracking*

Magic Number	Description
P2	ASCII PGM (greyscale)
P3	ASCII PPM (colour)
P5	raw PGM (greyscale)
P6	raw PPM (colour)

Table 3.2: *Supported input file formats.*

third dimension to the image structure: greyscale images have 1 channel and RGB images, for example, have 3. Access functions allow the 3 dimensions specifying the frame size and the intensity value at a specific 3D coordinate to be returned during processing. The latter value may also be reset during the life of a *VS_frame* object, whereas the others are fixed.

Class *VS_frame_io* provides i/o facilities for a frame, which are not included in the *VS_frame* class as most such objects do not require that functionality. Objects of this class allow input and output from/to files of the formats given in table 3.2.

The *VS_frame* and *VS_frame_io* class together provided the basic format required as a base for the inputs and the principal intermediate outputs of the system. These are to be in image-like format which is consistent with both standard practice in vision research and evaluation the context of semi-automated surveillance (see section 3.3.3).

The initial design work pre-dated the inception of the Vision Systems Code Library and a separate class was originally developed to define image-like entities. This *VS_image* class was retroactively modified to be a sub-class of the standard *VS_frame* class (appendix A, figure A.1) to allow compatibility and consistency. Similarly the *VS_image_io* class was converted to be a sub-class of*VS_frame_io*.

Initial design and development involved implementation of these classes, using single greyscale images stored as *.pgm* files to test the basic construction, input/output and arithmetic functionality of the classes. A pointwise operation function in the form of a simple edge detector was implemented at this stage to check the viability of the structure for applying vision processing operators generally.

The data members of the *VS_image* class itself include some more basic classes, *VS_point* and

Module	**Object**
Image loading	*VS_image_io*
Background estimation (1)	*VS_median : public VS_image_process* *VS_blur_image : public VS_image_process*
Background estimation (2)	*VS_mix_back* *VS_gauss_mix* *VS_gaussian*
Foreground segmentation	*VS_diff_frame : public VS_image_process*
Object growing/labelling (1)	*VS_object_map : public VS_image*
Object growing/labelling (2)	*VS_outline*
Object classification (1)	*VS_tracker* *VS_spline: public VS_image_process* *VS_model* *VS_pedestrian*
Object classification (2)	*VS_shapetrack* *VS_shape*
Object tracking	*VS_kalman* *VS_matrix*

Table 3.3: *Discrete objects used in the system variants*

VS_win, the first of which is a structure to hold a 2-D pixel coordinate and the second defining a rectangular area using two such points (appendix A, figure A.2).

The intermediate output of the system modules was primarily in the form of objects derived from the *VS_image* class so that the standard i/o interface could be used to store the results for review in the initial testing procedures.

The other objects required in the system correspond to simple or composite image processing techniques. For simple image processing functions, these were implemented from the base class *VS_image_process* in the VS library which has the virtual function *Process()*, used to process one frame and return the results of processing as another frame. Image processes derived from this class replace the *Process()* function with one that does a specific required task.

More complex, composite processes were used to define independent classes, which often correspond to modules within the system. A summary of the image processing objects, which is based upon the actual module choices made in section 4.2, is given in table 3.3 and their full class structure is specified in appendix A.

The *VS_kalman* class used in the tracking subsystem also comes directly from the VS library,

as does the *VS_matrix* class used in its matrix operations.

As noted above, the code is written in C++ to promote modularity, encapsulation, re-usage and for consistency with the VS library. At the outset of the project it was established that *CamOS*, the proprietary system used by all VideoBridge^{TM} units used what was described as a pseudo-C++ compiler. This point was not investigated in detail at the time, with the assumption being made that transfer of code from C++ to *CamOS* would be a relatively minor task. This turned out to be an error, as the *CamOS* compiler demands a code structure much more analogous to C code conventions in the structures used.

Conversion of C++ code to C is generally a non-trivial procedure: the class structures/interfaces, input/output functions and basic commenting rules are examples of elements of C++ which are not supported in C. The conversion process, while not challenging in terms of difficulty can be extremely time consuming in execution.

Only as late as the final project review, it was discovered that most C++ compilers can generate 'vanilla' C code from a C++ source as a first pass preprocessing stage. This can be processed by the CamOS compiler after relatively little additional work. This option was not appreciated until it was too late to take advantage of the potential avenue for accelerated conversion.

This error had significant ramifications as it made the proposed transfer of the final proposed solution to the end-use platform a time-intensive endeavour. This, coupled with the need to ensure that all code only requires fixed point arithmetic processing (see section 4.4.4.3) put the final transfer outside the time-scale of the current project.

3.9 Summary

The rationale underlying the construction of a modular surveillance system has been presented in terms of the distributed processing platform to be used and the intended nature and scope of the application to be implemented. The characteristics of the processing problem have been set out leading to a statement of the overall research goal.

The system design strategy has been discussed in terms of an initial top-down decomposition of the problem, followed by a bottom-up approach to designing the resultant modules. The general points to consider in the selection of candidate solutions for each module have been determined and the individual modules have been specified.

The next step, to apply the selection procedures for each module to choose the specific candidate solutions and to explore their implementation is presented in Chapter 4.

Chapter 4
A Surveillance Solution with Interchangeable modules

4.1 Introduction

Chapter 3 examined the overall design of a surveillance solution suitable for a specific system, directed toward a particular application. First, a top-down approach was taken to produce a specification in terms of the appropriate operational modules from a design perspective. Then object oriented design principles were employed bottom up to specify the meaningful objects for each module in the system, their attributes and their properties.

With the modular units so defined, the next key step in the design process is the selection of the best *candidate solutions* to implement as modules for the system performance characterisation. For each module, a choice must be made as to the most suitable approaches to be implemented within the context of the current system and application. The modular decomposition of the system is specified in section 3.7 and the details of the chosen solutions will be given in the following sections.

For each module implemented in the test system, a sufficient theoretical grounding will be given to justify the approach's applicability within the current application. The level of detail presented with respect to the implementation will be enough to allow the control structure to be understood and for the approach to be re-implemented in a similar form.

The correct functioning of each module must be established before integration into the final system and brief details are given of the format of these basic tests. The results of these initial runs are also used in setting *tuning parameters* for the modules to optimise results on the test data.

4.2 Choice of candidate solutions

In terms of suitability to the task in question, each possible candidate solution should be reviewed in the context of available knowledge of the system and application. The problem specifications both in terms of platform constraints and performance requirements are as set out in full in Chapter 3. These factors must be taken into account both when developing an approach 'in-house' and when evaluating the suitability of implementing a method based upon a previously published approach. In the latter case, the existing work on the approach should provide the subjective, qualitative results (perhaps in image-like format) which will serve as an initial basis for selection.

The approach suggested by this re-implementation can only draw on the original work for the seed of a solution. While they will share elements of a common theoretical background, the new implementation will need to be constructed from first principles to reflect the requirements and limitations of the final platform and so both the solutions will be essentially novel. This reinforces the fact that the published results can only serve as a qualitative guide to selection.

In this project, there are further constraints to put upon the selections for candidate solutions. Not only must the task-based constraints be fulfilled, but it is also desired that the comparative results be of interest in and of themselves. Little work has been carried out in the vision community on performance characterisation of alternative solutions on common data (see Chapter 5). The results will be most useful if the comparisons made during the evaluation process give information of more general value on the abilities of approaches which represent quite distinct approaches to solving individual problems.

There are then a further group of targets to consider within the selection process at a particular module:

- The theoretical basis of and assumptions inherent in candidates for a particular model should be considered, mindful of opportunities for interesting comparison.
- If there is a commonly used approach, this should be evaluated against (a) novel method(s).
- Approaches with contrasting levels of general complexity should be compared to evaluate the benefits of that complexity.
- New/interesting published approaches should be investigated to confirm their reported results.

- Any approaches the implementation of which for the final system would be sufficiently different from the extant work should give interesting and novel results.

A final additional consideration motivating the choice of the number and type of the candidate solutions is the novelty of the proposed evaluative process itself. As discussed in full in Chapter 5, there is currently no accepted procedure for making comparative qualitative performance characterisations of the type which will be valuable in assessing the system variants here.

In the absence of this, not only are the results of the evaluation important but also results on how well the proposed evaluation process itself assists in making a final determination of relative abilities. This suggests first that at least one approach for each module should be one where clear pre-existing qualitative performance appraisal exists. Secondly, the number of candidate solutions considered should be fixed at the minimum compatible with the development aims so that the exploration of of the evaluation process is simplified.

The exact consideration process in selecting the candidate solutions for each module are discussed separately in the section pertaining to that module.

4.3 Image capture and preprocessing

In the final application, the input data which will be provided to the system which serves the intended application will be uncompressed digital images. The specifics of the image format are described in Chapter 3, and the interface with the final platform is an issue to consider in detail when the system migrates from the test platform. This interface will take advantage of extant modules available for video stream conversion, modified as appropriate to deliver images in the *VS_image* format discussed in the previous chapter.

The steps constituting preprocessing will not be subject to methodical variation then, as their detail will not have impact on the final implementation. The choice of the procedures for supplying the input data to the first variable module will be linked to the planned testing procedures and the format in which raw data will be supplied.

4.3.1 Operational test data collection

The details of the data capture process used during the main performance characterisation testing phase are described in full in Chapter 5.

During the implementation phase of the modules sufficient input data was needed to test the basic function of the modules and to allow various internal parameters to be tuned optimally (see individual sections below). For this procedure, the test data required varies between procedures from numerical input to a full (real or simulated) sequence. At each stage of development, testing was carried out on the stand-alone modules and also incrementally on the sub-system variants implemented to that point.

For tests on image sequences to evaluate basic functionality, it is important to have a stored test sequence to allow the direct comparison of results and the repetition of tests. This is needed both in obtaining the correct core functions and in setting the tuning parameters of each to optimise stand-alone performance. Test image sequence data was obtained from the net (http://www.openvms.compaq.com/freeware/FREEWARE40/MPEG2PLAY11B/TENNIS.M2V) in the form of a short standard test sequence. The sequence used was *'tennis.m2v'* (see figures 4.2 and 4.3), an eight frame sequence of size 704x576 pixel frames sufficient to test the running and basic functionality of the program modules.

This file is in the MPEG-2 format which is a generic method for compressed representation of video sequences using a common coding syntax (section 4.3.2). The primary application for which MPEG-2 was developed was the all-digital transmission of broadcast TV but it is now used in many other areas including video conferencing and the compact storage of video sequences. The IndigoVision system uses H.262 coding for the transmission of video sequences, which has a common syntax with MPEG-2.

The MPEG-2 coding process involves colour space compression, motion estimation, DCT conversion, quantisation and Huffman encoding: the important point to note is that the storage space required is reduced by using a series of approximation processes. The overall result is a *lossy* compression of the sequence, one where the exact details of the original data can not be recovered precisely. Further, *artifacts* may be introduced during the process i.e. features may occur in the decoded images which do not correspond to real features in the original sequence.

By choosing to use the MPEG-2 format to conserve memory space it was also possible to take

advantage of the existing control and output options available within the context of the decoder (see section 4.3.2 below). The fact that storing images in the MPEG-2 format is lossy and may introduce artifacts during the encoding/decoding process has not been neglected and may indeed affect the absolute results of evaluation.

During the initial testing, the maximum achieved processing result is not under evaluation: what is of concern is the correct overall functioning of each module. The tuning of the parameters too is important, but the most appropriate settings for threshold levels, acceptance ranges, sensitivities etc. will not be affected by artifacts in the image data.

Due to the noted effects that using MPEG-2 files as input has on the absolute results, an alternative was needed for the qualitative testing procedures (and for the later stages of operational testing: see section 4.3.3 below). The choice of using a sequence of individual binary *.ppm* files as noted in Chapter 5 meant that a separate control structure had to be developed to use in the test harness. It should be noted that use of individual JPEG files was discounted for the same reasons that MPEG-2 was not acceptable, but that the new JPEG2000 format, with its promise of a lossless compression ability would be an interesting option for future experiments.

4.3.2 MPEG decoding

To work from MPEG-2 files, the first module required in the system was an MPEG decoder. The standard format of decoder takes a *.m2v* file as input and accepts control-line specification of the output type which can usually include separate or interleaved YUV components, Truevision TGA, PBMPLUS PPM and X11 display. It would be inefficient to perform the initial step of conversion to a sequence of still images which must themselves then be converted to *VS_Image* format (see Chapter 3). More useful would be an adapted decoder which includes an option to output a sequence of *VS_Image* objects direct to the next module.

Rather than devote time and resources to building a decoding utility from scratch or falling back on using a pre-compiled utility as preprocessor, an existing decoder available in the public domain as source code was modified to the specified requirements. Using 'off-the-shelf' components for which source code is available is advantageous in that the specifics of operation can be modified directly and re-modified to reflect development and other changes in the system. Also, any individual component routines (for example: the DCT algorithm here) can be made available if required in future processing steps.

The module obtained was MPEG-2 Video Decoder, Version 1.1, (Copyright June 1994 MPEG Software Simulation Group), a public release not optimised for speed in which emphasis is on correct implementation of the MPEG standard and simple program structure. The speed issue is not of undue concern as the decoding time will not be included in the evaluation phase measurements. The module was written in C, whereas the standard for the department's Vision Systems library software is C++, so the first required step was modification throughout the body of the programme to convert to C++.

The overall control structure of the MPEG decoder involves first opening the *.m2v* file and the reading of its header information which specifies sequence length and the coding parameters. A loop is initiated for the indicated number of frames to be decoded and after decoding each frame it is stored to file in the specified format.

In the modified decoder, the control line option specification is disabled and an additional output option integrated for 'storage' in the *VS_image* object format, which option is set as the default for the procedure. A call to a *FrameProcess()* function is included in the loop after the creation of the *VS_image* object. This function will be modified during development to correspond to the process(es) under test at that time.

After the specified number of frames in the sequence have been decoded/processed, the frame loop ends and the MPEG-2 file is closed. At this point, any further processing on accumulated data from the sequence overall can be performed by incorporating further process calls prior to the decoder routine terminating.

Open MPEG-2 file
Read file header, set up loop
While (frames to be decoded)
    Decode data for current frame to a *VS_image* object
    Store original frame (if required)
    Call *FrameProcess()* to execute processing on current *VS_image* object
    Store processed output
Close MPEG-2 file
Perform any processing required on the sequence data overall

Another option considered at this stage was to process the **coded** MPEG stream directly, moving the processing problem into a different data domain. A *VS_MPEG_frame* object could be created by modifying the decoder at an earlier stage in processing. This would lead investigations in a quite different direction to the initial intention (of investigating processing on image-like entities).

Although an interesting possibility, this option was discarded as it would constitute a step away from the anticipated real world application. Although in some situations image sequences may be stored in MPEG format in the final application, developing an approach assuming this would be restrictive. The process would be tied into this additional stage of processing which may not be required in most instances.

In summary, the MPEG decoder was integrated with the basic *VS_image* routines to enable generation of the desired stream of image-like entities. Once this sequence of *VS_image* objects is available, the image processing proper can begin.

4.3.3 Control structure when using *.ppm* files

As the cumulative subsystem complexity increased, the use of an MPEG file as the data source became less attractive. Its advantages were low storage overhead, the ability to call the novel image processing routines from within the decoder control cycle and the integral ability to store image-like results in the variety of formats available in the standard decoder.

Set against this, the test runs had to begin from the coded MPEG file for every subsystem test: the decoding overhead, although not to be considered in overall evaluation nevertheless took significant processing time, slowing down the rate of development. Also, the option of storing the results of one processing step in a lossless format to use as input for testing the subsystem from that point was not available if only MPEG source data was accepted.

Accordingly, from the point after development of the object growing module, the test system was altered to accept input data in the form of a sequence of individual binary *.ppm* files, using the standard input/output procedures for *VS_Image* objects. These options are inherited from the parent *VS_Frame* class (see Chapter 3) and allow input/output as binary or ASCII *.ppm* or *.pgm* format files

Background representations, difference maps and object maps could then be stored as image-like entities and used as input for testing later stages of processing in isolation. This made it possible to create artificial images to test the operation of modules as stand-alone units on quantitatively known images.

As noted, a separate control structure had to be developed to use in the test harness now that the control flow of the MPEG decoder could no longer be utilised. The format used followed the

specifications of the department's guidelines for writing Vision Systems code and the overall control flow is as follows:

(set t_1)
Specify global variables
Call constructors of required objects
Set flags to specify module choices
Perform conditional initialisation based on module choices
(set t_2)
For the specified number of frames
    (set t_3)
    Load current frame's file to a *VS_image* object
    Call *FrameProcess()* to execute frame-wise processing on current *VS_image* object (and set t_4,t_5,t_6 etc.)
    Store processed output
    Increment frame counter
Perform any processing required on the sequence data overall
(set t_7 and report timing/memory results)

The bracketed elements correspond to modifications to include timing and memory estimation routines for performance characterisation (Chapter 5).

4.4 Background estimation and foreground segmentation

4.4.1 Background representations

4.4.1.1 The significance of background representations

As discussed in Chapter 2, efficient and robust construction of a dynamically evolving background estimate is a vital first step for versatile scene analysis. While some environmental knowledge may be significant in later stages of interpretation, in the initial stages we desire to, as far as possible, remove the effects of as many environmental variables as possible, and to do this they must be (implicitly or explicitly) modelled in the background estimate.

Segmentation of foreground objects, the differentiation of them from the background, may be integral to the background estimation or can be implemented as a separate process. This must usually include the first levels of noise elimination and the removal of potential objects characterised as 'not significant' by some heuristic rules.

Before considering the choice of candidate solutions for this module, it is worthwhile to address

the meaning and desired characteristics of a background for surveillance purposes.

- The most basic principle is that the background should register *permanent*, essentially unmoving objects in the scene and identify them as background components. The simplest case would be a scene of rigid, immobile objects under constant lighting conditions: in the absence of any moving objects the background would be the scene itself. The task of a background estimation approach is to analyse a dynamic scene and in some way to allow moving objects to pass through the scene without significantly affecting the knowledge of the underlying background.

- New objects placed 'permanently' in the scene should be incorporated into the background representation (and those removed should similarly be replaced by the background which they formerly occluded). This dynamic behaviour must be balanced against the ability to distinguish moving, impermanent objects and so some latency (a delay between the object appearing and its being incorporated into background) is usually anticipated.

- The background should be robust with respect to 'slow moving' objects: these should not be incorrectly incorporated into the background. There is usually a trade-off to be made between the ability to correctly exclude objects moving below a certain velocity and the latency with respect to registering new objects.

- The estimation approach should be able to cope with a variety of backgrounds, including those which develop dynamically over time. Apart from the introduction and removal of new objects as noted, this will include the significant case of lighting variation. To take the example of exterior scenes, the natural lighting varies over a 24 hour cycle, changing the average contrast of the image and the pixel values of points corresponding to the same scene elements.

  This diurnal lighting change is normally viewed as a slow variation, compared with the anticipated time scale of changes due to moving objects. Such contrast changes will also occur on shorter time scales due, for example, to the passage of a cloud in front of the natural light source or the change of artificial lighting affecting the scene.

- The approach should be able to cope with the effects of environmental noise, for example rain or snow in an outdoor environment, or the effects of reflections on intervening glass surfaces.

- Ideally, the approach should in some way be able to cope with motion *within* the background. The examples usually cited are of oscillatory motion due to the motion of vegetation in the wind and of specularity effects on the surface of water in direct sunlight. If the approach cannot adapt to recognise this motion as an element of the background itself, the effects are likely to be registered as foreground objects and it may be possible to eliminate or mitigate the effect in the noise handling procedures.

- The nature of a background representation will vary dependent on the variability allowed in the camera's intrinsic and extrinsic parameters. Where a camera is allowed degrees of freedom in respect of translational motion, pan, tilt or zoom, the background representation problem becomes more complex. In this situation, information on the changes in these parameters could be combined with a representation of the background which covers the full range of their allowed values to generate the appropriate estimate for the current settings.

  The case of a stationary camera with fixed intrinsic parameters requires only a representation for a single invariant scene space and is the subject of most work in the area. If a process developed for this situation is used where camera parameters may vary periodically, there would be a latency period after the change as the background estimate is updated. In effect, this is equivalent to removing all background objects or replacing them with new items thus the latency which echoes the object introduction/removal case.

There are some further desired characteristics more appropriately identified with the foreground segmentation approach:

- Foreground extraction can be with respect to a dynamically evolving background as detailed or simply with respect to an unmodified earlier scene image.

- Segmentation is usually conducted on a pixel-by-pixel basis, with objects constructed from these initial results using a subsequent object growing module (see section 4.5).

- The approach should be able to deal with 'camouflage' effects in extracting objects. In image terms, this effect is the result of a low contrast between the foreground and background, due to poor lighting or similar colour levels.

- Balanced against the first consideration, some small changes with respect to background should not be interpreted as foreground objects, but used to update the background value.

- Noise effects which are not interpreted as a valid part of the background should be discounted. The definition of what is noise as opposed to an 'interesting' object will be task dependent to some degree and so this functionality should ideally be variable.
- There should be a fast response to new foreground objects appearing within the scene.

4.4.1.2 Choice of candidate solutions

To provide a contrast between a mathematically complex and a less complex approach, the first background estimation approach selected was one based on the most intuitive background representation, the long term average image. As far back as the 19^{th} century, this was used in photographic effects, exposing a film over a long period of time to capture long-term effects while omitting relatively fast moving objects.

A development of the idea is to weight the contributions of the time series of values observed at a point so that their relative contribution decreases exponentially over time. This gives a background estimation which can adapt dynamically over time to encapsulate, for example, the effects of changing lighting in a scene.

Conceived in 1995, the *Median average background representation* has been shown [25] subjectively, by use in surveillance applications similar to that under examination to give good results and is quoted as being able to provide functionality in respect of most of the quoted characteristics.

This approach generates an image-like background and requires a separate segmentation method. The most commonly employed partner method is provided here, direct frame differencing between the current frame and the background image. Both this and the background estimation can be implemented directly based on published details with no significant modification.

For an alternative method, the approach selected was based on a relatively new method solution published in the literature [50]: a *Mixture of Gaussians* background representation. The published results indicate qualitatively superior performance, especially in modelling backgrounds with significant oscillatory behaviour.

A direct re-implementation of the approach is not possible in this case due to the specific requirements of the system and several novel modifications were required to give a useful variant method.

This approach integrates background estimation and foreground segmentation and so does not require a supplementary module for the second task. However, to give results which are comparable with the median background approach and also in a suitable form for use by the following module, interface output functions were needed.

4.4.2 Median background generation

The first approach considered for constructing a background which captures the characteristics as specified in section 4.4.1.1 involves the building of a *time-averaged median background* image. In this approach, an image-like background is generated with the intensity value at each pixel position being derived from the weighted median average of the values seen there over previous frames.

The approach was first developed in 1995 [28] for the purpose of detecting piglets in an overhead scene of a livestock storage area. The key similarity with a surveillance scene is that frame differencing can be performed to detect objects, but only where a current frame and reference background can be compared. In both cases, it may not be possible to capture an image of the uninterrupted background and even where this is possible the background may evolve over time.

Conceptually, the pixel setting is intended to approximate the underlying background, being incremented/decremented toward the current frame value, which is the most recent information and which should therefore be the most significant: this allows the system to develop dynamically with a more rapid response time.

To initialise the background, a basis image is constructed by simply copying the first frame, which may contain foreground objects occluding some real background features. There will be an initial equilibration period as the update process gradually eliminates these objects in favour of the 'true' background values. For each subsequent frame the intensity values at each pixel position in the background are dynamically updated over the life-span of the process using the formula:

$$A_t = wA_{t-1} + u(I_t) \tag{4.1}$$

where:

$$u(I_t) = \begin{cases} -1 & : \quad I_t < A_{t-1} \\ 0 & : \quad I_t = A_{t-1} \\ +1 & : \quad I_t > A_{t-1} \end{cases} \tag{4.2}$$

and:

A_t is a pixel value in the average image at time t

w is the weighting factor

I_t is a pixel value in the current frame image

Without the weighting element, this would result in a pixel's value converging at a level where half of the historic values seen at that point were higher and half were lower, the *simple time-averaged median* . Inclusion of the weighting factor to give a *weighted time-averaged median* allows the value to be biased in favour of more recent values, giving a faster response to variations in the background.

The background value is incremented/decremented by a constant amount irrespective of the magnitude of the deviation of the current observed value from the previously generated background. This makes the approach less sensitive to outliers (values separated from a distribution mean by a large amount compared to its standard deviation). Thus unwanted change in the background caused by the passage of an object of greatly divergent colour is less than, for example, a weighted mean average background.

It is the frame-wise update process which underlies the latency in adjusting to object introduction/removal and an *initialisation latency* as the background is refined over several frames to exclude any moving objects in-frame when the unit was activated.

Use of this time-averaged median background image as the reference for object detection can not only take account of lighting, but also counter the effects of changes in the environment caused by objects being introduced and left (e.g. cars parked, snow cover). Where a sub-class of such changes are considered significant by the user, (perhaps a suspicious package being left behind) steps can be taken to modify the approach to give alarm signals in these instances.

The control structure for the time-averaged median background are relatively straightforward:

For every pixel in the background image

    Get the corresponding pixel value from the current frame (I_c)

If this is the first frame
    Set the background pixel value (I_b) to (I_c)
else
    Get (I_b)
        If $I_c > I_b, I_b = I_b + 1$
        If $I_c < I_b, I_b = I_b - 1$

4.4.3 Difference map construction

Immediately before the background representation is updated from the current frame, a *frame differencing* approach is applied to execute foreground segmentation. In this method, the value of each pixel in the current frame I_{c_t} is compared with the value of the corresponding pixel in the background representation $A_{c_{t-1}}$, for each of C colour channels independently:

$$\sum_{c=1}^{C} \Delta_{c_t} = I_{c_t} - A_{c_{t-1}} \tag{4.3}$$

where Δ_{c_t} is the pixel value of channel c in the difference map at time t. The difference map is an image-like representation of the differences between the two source images.

The results of the single channel differencing are combined using the standard JPEG formula as a first stage filter for noise.

$$I_{xy} = 0.59G_{xy} + 0.11B_{xy} + 0.30R_{xy} \tag{4.4}$$

where I_{xy} is the greyscale intensity
G_{xy} is the green channel intensity
B_{xy} is the blue channel intensity
R_{xy} is the red channel intensity
The idea here is that taking the difference for each colour channel separately and then combining these will 'even out' any channel specific variations in what is effectively a colour-channel blurring procedure.

Gaussian low-pass filtering is applied to the combined difference image to further reduce noise, mathematically:

$$G(x, y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}} \tag{4.5}$$

where $G(x, y)$ is the filtered result

σ is the standard deviation of the Gaussian distribution

x, y are the coordinates of the centre of the distribution

This filtering is implemented by *convolution* with the 2D *kernel* (or *mask*):

K_{11}	K_{12}	K_{13}
K_{21}	K_{22}	K_{23}
K_{31}	K_{32}	K_{33}

with the K_{ij} set to integer values:

$$K_{11} = K_{13} = K_{31} = K_{33} = 1 \tag{4.6}$$

$$K_{12} = K_{21} = K_{23} = K_{32} = 2 \tag{4.7}$$

$$K_{22} = 4 \tag{4.8}$$

These values correspond to the most commonly used 3x3 Gaussian mask [17].

The process of convolution involves the placement of the mask at each pixel point in the image in turn. The value at that point is then replaced by the sum of the products of the mask entries with the pixel values at the corresponding points in the unblurred image. Mathematically:

$$G(x, y) = \sum_{i=1}^{m} \sum_{j=1}^{n} I_{(x+i-1, y+j-1)} K_{ij} \tag{4.9}$$

the principle here is that this filtering will 'even out' the spatial noise in the image by replacement of values with a weighted average of the values in the immediate neighbourhood. The spatially small (typically single pixel) random noise effects will be strongly modified in this process such that they may no longer be detected as object pixels in the later thresholding process. There will be a lesser effect on large objects whose values will tend to be reinforced by similar neighbouring values (although there will be more effect at the edges).

The resultant filtered, single channel, greyscale difference image is *thresholded* to give a 'trinary' image. For a threshold value T, pixel values in a band about 0 defined by $\pm T$ are set to zero,

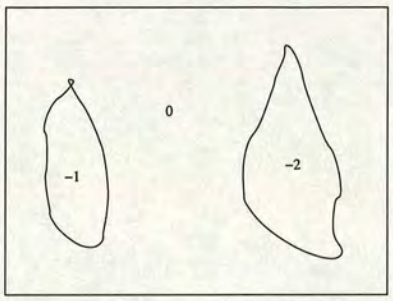

Figure 4.1: *An example of a trinary difference image. The leftmost object was registered at values below the threshold band and is set to* -1. *The rightmost object was registered at values above the threshold band and is set to* -2. *All other points fell within the threshold band and are set to* 0

those below the band are set to -1 and those above are set to -2.

$$I_{T_{i,j}} = \begin{cases} -1 & : \quad I_{i,j} < (-T) \\ 0 & : \quad (-T) \leq I_{i,j} \leq T \\ -2 & : \quad I_{i,j} > T \end{cases} \tag{4.10}$$

Thus there is a two-part foreground of non-zero values corresponding to potential moving objects against a zero-value background. If an actual moving object is brighter than background, -1 will represent its position in the current frame, -2 in the previous frame. Conversely, if the object is darker than background, -2 will represent its position in the current frame, -1 in the previous frame. A simple example of such a trinary difference map is illustrated in figure 4.1.

In choosing the threshold levels, a balance has to be struck between the amount of noise which can be subtracted and the system's ability to give a consistently contiguous object image. For our purposes, the bias of the thresholding is toward 'foreground' *(conservative thresholding)* to promote object continuity.

Where an extracted object is smaller than a threshold number of pixels in size (20 pixels used here) it is discarded on the grounds that it is both likely to constitute noise and also on the practical grounds that it will be of little use for accurate shape estimation or spline fitting. This should be noted as a lower boundary to the scale invariance of the system: below this size otherwise potentially valid objects will not be recognised.

The output of the combined background estimation and foreground segmentation processes is a sequence of trinary difference maps (all *VS_Images*), each 'blob' therein corresponding to a

hypothesised moving object is the current frame.

The command structure for both frame differencing and filtering too are relatively straightforward:

For every pixel in the current colour *VS_image*
    Convert the 3 channels of colour information into a single value using the JPEG formula
    Store as a greyscale *VS_image*
For every pixel in the greyscale *VS_image*
    Convolve with the Gaussian kernel
    Store resulting value in the corresponding position in a temporary *VS_image*
Copy the temporary image values into the greyscale *VS_image*
For every pixel in the blurred greyscale *VS_image*
    Compare the value (I_c) with the threshold value T
    If $I_c < -T, I_b = -1$
    If $I_c > T, I_b = -2$
    else $I_b = 0$

Implementation points The trinary map values corresponding to foreground were set as negative values to assist in distinguishing them from labelled object pixels (marked at positive integer values) in a later stage of processing. Trinary maps are thus unsuitable for direct display as image like entities, as negative values have no meaning in this context. A conversion procedure to display the foreground components at suitable positive values must always be incorporated if a visual representation is required.

In the convolution stage, special boundary conditions are implemented at the edge and corners of the frame. For all such operations using the values of neighbouring pixels, special cases must be applied at the boundaries where the normal algorithm looks for values of pixels outside the image. In this implementation, the special cases project 'mirror' values for pseudo-pixels outside the boundary.

4.4.4 Mixture-of-Gaussians pixel model

More complex alternatives to viewing the background as a long term average exist which involve modelling the individual background pixels explicitly. The median average background approach, along with all others which implicitly or explicitly model a pixel using any single distribution make the assumption that its value is due to a single static surface. In this case, changes in the pixel's value will be due to:

- An object occluding the background surface
- The removal of the original background surface
- A change in environmental conditions (lighting/visibility)

To move away from this simplistic assumption, it is useful to consider the variation in values observed at a pixel $p = (x_p, y_p)$ as a time series. At time t_p, the data upon which a prediction of the pixel's value can be constructed are the pixel values observed at that point historically:

$$\{I_0, I_1, ...I_{t_p-1}, I_{t_p}\} = I(x_p, y_p, t) : 0 \leq t \leq t_p \tag{4.11}$$

where I_t is the intensity at time t.

In these terms, the process involved at this pixel in generating, for example, a mean weighted average background is:

$$B_{t_p} = \sum_{t=1}^{t_m} \alpha^t I_{(t_p - t)} \tag{4.12}$$

where:
B_{t_p} is the value of the background at this pixel at time t_p
t_m is the number of recent pixels used in building the background
α is the (fractional) learning rate

In proposing an alternative approach, [50] notes that the long term average models share failings in that they are not robust to scenes with many objects and tend to allow slow moving objects to become partially incorporated into the background too readily. This is because there is a single background value at every pixel which is modified to some degree by every non-background object which passes there.

That alternative involves modelling the observed time series at the pixel by a probability density function (PDF). The PDFs for all pixels at the current time can then be considered as a whole and a heuristic decision made as to foreground/background classification

In reviewing the decision of an appropriate basis function to use in modelling the intensity values, it is useful to go back to the realities of the observed scene. Considering an idealised scene with constant lighting and no moving objects, the representative time series for a pixel would be a static repetition of a single value (scalar for greyscale images, a 3-vector for RBG/YUV). The static value would correspond to the colour and brightness of the world object corresponding to the pixel's coordinates in the image plane.

Moving away from the ideal and considering noise due to acquisition and processing, one would expect variation around the static value over time. Assuming multiple independent factors cause the noise, the variations would be random, individually small and with an effect over time which averages to zero. The Central Limit Theorem states that the best representation for modelling such a process is by a *Normal* or *Gaussian* distribution.

Given this, standard statistical methods [104] show that the PDF, η, giving the best model of the observed values is:

$$\eta(x_t, \mu, \Sigma) = \frac{1}{(\sqrt{2\pi})^n \sqrt{det(\Sigma)}} e^{-\frac{1}{2}(x_t - \mu_t)^T \Sigma^{-1} (x_t - \mu_t)} \tag{4.13}$$

where:
x_t is the pixel value at time t
μ is the mean of the Gaussian (the static value expected)
Σ is the covariance matrix of the Gaussians (the second order moments: in one dimension, the variance)
n is the dimensionality of the image (1 for greyscale, 3 for colour)
det is the determinant of a matrix

This Gaussian model of pixel value can be extended by using each new measurement of intensity to update the values for μ and Σ to give an *adaptive Gaussian model*, able to handle variation of lighting over time.

So far, the calculations to be applied have been increased in complexity with little evident in terms of theorised increased performance over the time average background approaches. What using a distribution in this way leads us to, however is the ability to consider a newly observed pixel value probabilistically in terms of previously seen values. We can form conclusions as to how likely it is that the observed intensity is within the expected distribution of values as

opposed to being due to a new element.

This attribute is of great value in practical applications where it is usually not possible to make a uni-modal classification of a pixel as belonging to a single background or being part of an extant/new foreground object. In real,world scenes a 'background' point may vary between values corresponding to, for example, **road/leaf/branch** as a local bush is blown in the wind. The pixel corresponding to this behaviour would thus oscillate between at least three values even in the absence of a foreground object or light variation.

Considering the occurrence of values as a distribution, intensity against time, this situation would be reflected by a multi-modal pattern which cannot be satisfactorily represented by any single simple function. Here the value of having a *mixture* of distributions available for representing each pixel becomes evident. A new value can be compared with the mixture and either classified as belonging to an extant distribution or being part of a hitherto unseen one. An additional advantage is that a 'primary' background distribution may not be lost if an object temporarily becomes part of the background. The primary distribution remains in existence until it is the least probable one when a new object appears and so can be quickly re-incorporated when the temporary object moves on.

4.4.4.1 Calculation of mixture model

The mixture which best represents the distribution at any point in time can be considered to be that linear combination of basis functions (Gaussians here) whose product of likelihoods is maximised when taking into account all observed values. We can define the mixture model at time (t) as $P(I_t)$:

$$P(I_t) = \sum_{i=1}^{k} w_{i,t}\eta_i \tag{4.14}$$

where:

$w_{i,t}$ is the ith mixing coefficient at time t.

η_i is the ith distribution.

k is the number of Gaussians used to represent each pixel.

Then the product of likelihoods (λ) is:

$$\lambda = \prod nP(I_t^n) \tag{4.15}$$

A strict algebraic solution of the maximisation requires computing the derivatives with respect to the parameters (both mixing weights and each Gaussian's parameters) and using these in a non-linear optimisation algorithm [104] . This complex constrained optimisation problem requires the solving of highly non-linear coupled equations which is prohibitively complex and time-consuming for this application. The preferred alternative is to apply a re-estimation technique based on the expectation-maximisation (EM) algorithm.

This algorithm can be summarised as:

Initialsation:

Set k Gaussians with means at a random step from a point in the data set.

Set the initial covariance matrices to the identity matrix and the initial weights to 1/k.

Expectation:

Use the Gaussian PDF generated by the current mixture to calculate a likeliness estimate for each point.

Maximisation:

Re-estimate the parameters using the results of the expectation calculations

Convergence:

Repeat expectation-maximisation cycle until the parameter values cease to change significantly.

Applied to our situation, initialisation will be incremental, in that a first distribution will be set up at the pixel's initial value (as above) and distributions will be added as needed up to the maximum.

Even using the EM algorithm, with re-estimation required for each pixel at each time step, the processing required would be prohibitive in an on-line real time application. We use a limiting case of the EM algorithm assuming spherical Gaussian basis functions with a common parameter σ to replace the covariance matrix $\Sigma_{k,t}$, where:

$$\Sigma_{k,t} = \sigma_k^2 I \tag{4.16}$$

and I is the identity matrix

This means we are making the simplifying assumption that, for the RBG case for example the same variance is observed in each colour channel. Although this is not going to be the case in reality, the computation saving is worth the loss in accuracy. Considering the limit of $\sigma \rightarrow 0$

[104] gives the *K-means clustering* approach which is implemented. This can be summarised (from [104]) as:

1) Compare the current pixel value, I_t, with the extant distributions for that pixel.

2) If:

$$(\mu_i - (2.5)\sigma_i) \leq I_t \leq (\mu_i + (2.5)\sigma_i) \tag{4.17}$$

for distribution i, we have a match and mark the distribution to be updated.

Otherwise, we have no match. Replace the distribution with the lowest probability with a new distribution such that:

$$\mu_{new} = I_t \tag{4.18}$$

$$\sigma_{new} = \sigma_{t-1}^{max} \tag{4.19}$$

$$w_{new} = w_{t-1}^{min} \tag{4.20}$$

3) Adjust the weights of the old models using:

$$w_{k,t} = (1 - \alpha)w_{k,t-1} + \alpha(M_{k,t}) \tag{4.21}$$

where:
α is a *learning rate* defined by the inverse of the time constant which defines the rate of change of parameters. $M_{k,t}$ is 1 for models marked as matched, 0 otherwise.

4) Re-normalise the weights so that:

$$\sum_{i=1}^{k} w_{i,t} = 1 \tag{4.22}$$

5) For any models marked as matched, update the parameters using:

$$\mu_t = (1 - \rho)\mu_{t-1} + \rho I_t \tag{4.23}$$

$$\sigma_t^2 = (1 - \rho)\sigma_{t-1}^2 + \rho(I_t - \mu_t)^T(I_t - \mu_t) \tag{4.24}$$

where :

$$\rho = \alpha\eta(I_t, \mu_k, \sigma_k) \tag{4.25}$$

This is the application of standard learning rules, taking the new pixel value as the new sample. The overall result of the process to this point is an up-to-date *Mixture-of-Gaussians* model for each pixel in the image.

4.4.4.2 Classification of models

The final step in the segmentation process involves the classification of the Gaussians as background/foreground. A heuristic rule needs to be applied to differentiate between the two, based on logical inferences about real world scenes. We need to establish some distinctive characteristics of background pixels which can then be used in selecting the distributions.

Firstly, by their very nature, we expect distributions relating to background to be present for longer that those relating to new *and/or* moving *and/or* transient objects. Thus, considering equation 4.21 for the mixture parameters we would expect that, at any time, Gaussians corresponding to background values will, on the whole, have a greater value for w than models which do not. Also, we expect those relating new/moving/transient objects to have a greater variance due to the changes in the object over time.

So, the likelihood that a pixel corresponds to background has an inverse relationship with σ and a direct relationship with w. It seems reasonable, then to consider the value $\frac{w}{\sigma}$ which increases both for increase in w *and* decrease in σ. All other things being equal, we would expect Gaussians with a high value of $\frac{w}{\sigma}$ to be most likely to correspond to background.

The process of distribution classification using this premise is:

- Sort the individual Gaussians representing the pixel in order of decreasing $\frac{w}{\sigma}$.
- Classify enough Gaussians in the ordered list as background, such that a pre-defined minimum proportion (T) of the mixture is classified as background.

Empirical selection of the parameter T to allow more than one of the constituent Gaussians to constitute background allows the system to handle a multi-modal background.

4.4.4.3 Implementation notes

Objects

The implementation of the *Mixture-of-Gaussians* segmentation approach uses three distinct objects:

- *VS_Gaussian* objects have a mean value, variance, current weight and flags which indicate whether they are currently matched and and/or classed as background.
- *VS_Gauss_mix* objects held an integer size and a 1-D array of that many *VS_Gaussian* elements.
- *VS_mix_back* objects have a 3-D array of *VS_Gauss_mix* object equal to the image size (area x number of channels)

Mixture size

Three Gaussians per mixture were used throughout the current implementation. The paper which first developed a similar approach [50] investigated between 3 and 5 as mixture sizes and the lowest of the range is implemented here to limit memory and computation requirements.

Initialisation

There was no information given as to choice of initialisation procedures for the Gaussians in the above paper. It was decided that each *VS_Gaussian* object would be initialised by setting the mean outside the range of possible image values (at 300), with a low variance (1). The first value observed at each pixel will be outside the acceptance range of the corresponding mixture's Gaussian distributions and so will form the basis for a new distribution centred on that value. The initial variance is set high (400, giving a standard deviation of 20) to allow a large range of acceptance about the first observed value. As new values are matched, the variance will equilibrate to a more reasonable value.

Colour matching

The match process is carried out for each match separately. The same Gaussian must be matched in each channel for an overall match to be reported.

System restrictions: fixed point arithmetic

A significant development of the published work was required to construct a similar method suitable for implementation on the final platform. On that platform, the application will be

restricted to using *fixed point arithmetic*, which is a major complicating factor in applying a procedure based around calculating probabilities under a Gaussian distribution.

There is no direct ability to calculate the exponent values central to the process nor the required square root values and calculations with probabilities, restricted to the range 0 to 1, are not manageable. Alternative solutions must be developed, keeping as close to the original methods as feasible and applying reasonable approximate methods.

Calculating probabilities for unmatched values On considering how to limit the need for the exponent calculations required in explicit calculation of these probabilities, a useful computation reducing solution was found. The probability (P) of a value varies monotonically with the separation of the value from the mean and is inversely proportional to the variance of the distribution:

$$P \propto S = \frac{(x_t - \mu)^2}{\sigma^2} \tag{4.26}$$

In evaluating whether or not a value matches with a distribution, we are assessing whether it falls within 2.5 standard deviations of the distribution's mean value, μ. Thus the match condition can be re-written simply in terms of separation as:

$$|x_t - \mu| \leq 2.5 \tag{4.27}$$

This considerably simplifies the calculation required in the matching process.

When a current pixel value does not match to an extant distribution, the Gaussian which has the lowest probability in the current mixture is replaced by a new distribution centred on the new value. In the original approach, the probabilities for each distribution were calculated and compared in making the replacement decision.

As the calculation requires only a comparison of probabilities, it is sufficient to instead compare the value of the scaled separation, S: the distribution which gives the highest value here is the least probable in terms of the current value.

These changes are **required** in the fixed point implementation, but also present a computational saving with no loss of accuracy or precision suitable for the original method.

Updating matched distributions As part of the distribution update process, a full probability

calculation is required. From our match criterion though, we know that we need only calculate e^{-x} for x which fulfills equation 4.27. This limit on the value of x makes calculation of a reasonable approximation to e^{-x} feasible using a lookup table and linear interpolation. In constructing a lookup table, values of e^{-x} are calculated for discrete x within the allowed range using floating point procedures and are stored in a 1D array. When an exponent value is to be calculated in the final system, the known values on either side are accessed from the table.

Using linear interpolation to project the desired value makes the assumption that e^{-x} is itself linear, which is not the case. However, exact correspondence to the Gaussian update formula is not considered vital to the functioning of the system: if the values are close enough together, an approximation good enough for use in updating the distribution can be obtained.

Calculation of the square root values using fixed point arithmetic utilises *Newton iteration* to extract an good approximation.

'Shift' arithmetic The restriction to integer arithmetic generates general problems in performing precise calculations and representing results, especially where intermediate results are restricted to the range 0 to 1. This problem will apply in each of the modules and a solution is vital for transferral of each to the final platform.

The solution for fixed point arithmetic is a process of *scaling* and *de-scaling* before, during and after calculations using the C++ *shift* operators. These operators are a fast way to execute multiplication and division by factors of two: the goal in their use here is too keep all results (final and intermediate) within the range which can be be represented on the system to acceptable precision. Simply, input is scaled up to the maximum amount that does not give an overflow of the register within the expected range of performance.

The organisation of the number of shifts to perform for each stage of the calculation involves knowing the allowed input range, and the effect that operations have on that range so that optimal shifting is performed in respect of each intermediate result. It is critical to keep track of the number of shifts which have been performed so that the final result can be restated correctly.

Loss of precision The use of look-up tables, linear interpolation, Newton Iteration and shift procedures introduce a potential loss of precision as the cost of having reduced computational complexity. Each procedure was tested in isolation against results of normal

floating point calculations to ascertain the degree of this loss: it was found that the results were accurate to a level of precision in excess of that of the image format. There will thus be no effective loss in precision.

Overlap of distributions

In the first test implementation, the *CheckSpread()* function was used after each updating or replacement to ensure that distributions' acceptance ranges for matching new values did not overlap. The new/updated distribution's acceptance range was 'cropped' at the interface with an extant range by reducing the variance accordingly. The problems with this method were twofold: firstly, the range reduction was symmetrical, and so an undesired 'mirror' reduction in acceptance occurred on the far side of the distribution's mean. More significantly, modifying the variance in this way distorts the modelling process, inhibiting the normal development of the distribution.

The *CheckSpread()* function was removed and the alternative solution was a modification to the matching procedure. The *TestMatch()* function was altered to check all distributions for a match with any new point, with the distribution giving the optimum value of S accepted as the matching distribution.

Choice of tuning parameters

α is the learning rate which specifies the latency in the update process and so how quickly the background responds to change. If this is set too high, unwanted slow moving objects will be incorporated into the scene, if too low then changes in the background will not be recognised on a timely basis.

T is the minimum proportion of data which must be attributed to background in the classification process. Where T is set low (comparable with $1/n$ where n is the number of Gaussians in the mixture), a unimodal background is likely. This will reduce reduce processing costs but will also sacrifice the characteristic improvements attributed to the approach. The higher T is set, the more modes it is likely will be included in the background representation, giving greater flexibility at the cost of increased processing costs.

α and T are the two tuning parameters quoted as significant in the paper on the original *Mixture-of-Gaussians* background method. In developing the current method, these parameters were set empirically during the functionality testing procedures. The criterion used in their initial setting was qualitative visual review of the output of the system on the test sequence. As the

sequence was quite short and without significant multi-modal behaviour, the settings were not fixed at these values immediately: they were finally fixed during later tests on a longer sample sequence (see Chapter 5).

A further significant parameter which needs addressing is n, the number of Gaussians in the mixture. The considerations governing the choice of this value are those noted in respect of T, the balancing of processing cost against representational flexibility. In this implementation, n was set at 3 from the outset: a 3-distribution is the minimum size required to offer the ability to represent a multi-modal (here bimodal) background plus foreground objects.

Finally, the criterion used to evaluate a match to a distribution is that the new value should fall within $\mu \pm 2.5\sigma$, that is within 2.5 standard deviations of the mean. This criterion gives a correct match for Gaussian distributed data in 95% of cases. As probability values in the Gaussian distribution approach zero asymptotically away from the mean, 100% confidence in matching is not possible. The choice of 2.5 standard deviations as the threshold is taken from the original approach and has a good basis in the use of 95% confidence intervals in statistics.

Output Configuration

The raw output of the module is a *VS_mix_back* object for each frame, with the *VS_Gauss_mix* objects on the 2D array each containing updated *VS_Gauss* objects with *MatchFlag* and *Bg-Flag* variables set appropriately. If the intended standardisation of module input/output is to be followed exactly, a supplementary module was needed to convert this to an image-like trinary difference map as produced by the *VS_median* procedures. The alternative is to relax the guidelines and allow modification of the succeeding module to recognise both forms of input, allowing future segmentation modules to give output in either format.

The latter alternative was chosen as the trinary difference map format has no functional significance to the *Mixture-of-Gaussians* background method or the system overall: conversion to this format as a matter of course would be an inefficient use of resources. To facilitate this, it is useful to set a flag to indicate which background module is used so that the appropriate input option in the object growing module is employed.

A more elegant and general solution to consider is to make all background generation classes subclasses of a generic abstract background building class. The subclasses could inherit important common attributes and succeeding modules would call on a member of the abstract class as input. The appropriate input option would be called depending on the specific subclass

of object received by that module.

4.4.5 Functionality testing

The first step in functionality testing is the inclusion of procedures for the image-like output of intermediate results within the implemented code. Even though these modules will not be used in the final implementation, care must be taken in their design and construction. If they are constructed poorly, misleading results as to the functionality of the module under test will be given, with repercussions throughout the design process.

For output as standard binary *.ppm* files, it is most efficient to generate results in the *VS_Image* format and to employ the standard i/o functions available to objects of that class (allowing binary or ASCII *.ppm/.pgm* files).

For the *Median average background/frame differencing* approach, the procedure required to display the segmentation result is conversion of the extant trinary difference map (itself a *VS_Image*: see class diagram in Appendix A) to a comparable difference map without the negative values which cannot be displayed. A simple loop routine is used to handle the three cases to convert:

For every pixel in the trinary difference map *VS_image*
    Get the current pixel value (I_c)
    If $I_c = 0, I_c = 254$
    else $I_c = I_f$

For the purposes of visual output, background pixels are set to 100, corresponding to a pale grey colouration. Foreground pixels are set to a value I_f, to give sufficiently discernible objects in the visual representation. An I_f value of 200 was suitable for this purpose.

Figures 4.4 and 4.5 illustrate the output format used for the segmentation result on the *tennis.m2v* sequence shown in figures 4.2 and 4.3.

Visual display of the background itself is trivial: the *Median average background* is a *VS_image* object suitable for storage as a *.pgm/.ppm* file suitable for direct display, and an example of this visual output for the *tennis.m2v* sequence is given in figures 4.6 and 4.7.

For the *Mixture-of-Gaussians* background representation, the procedure is rather more involved.

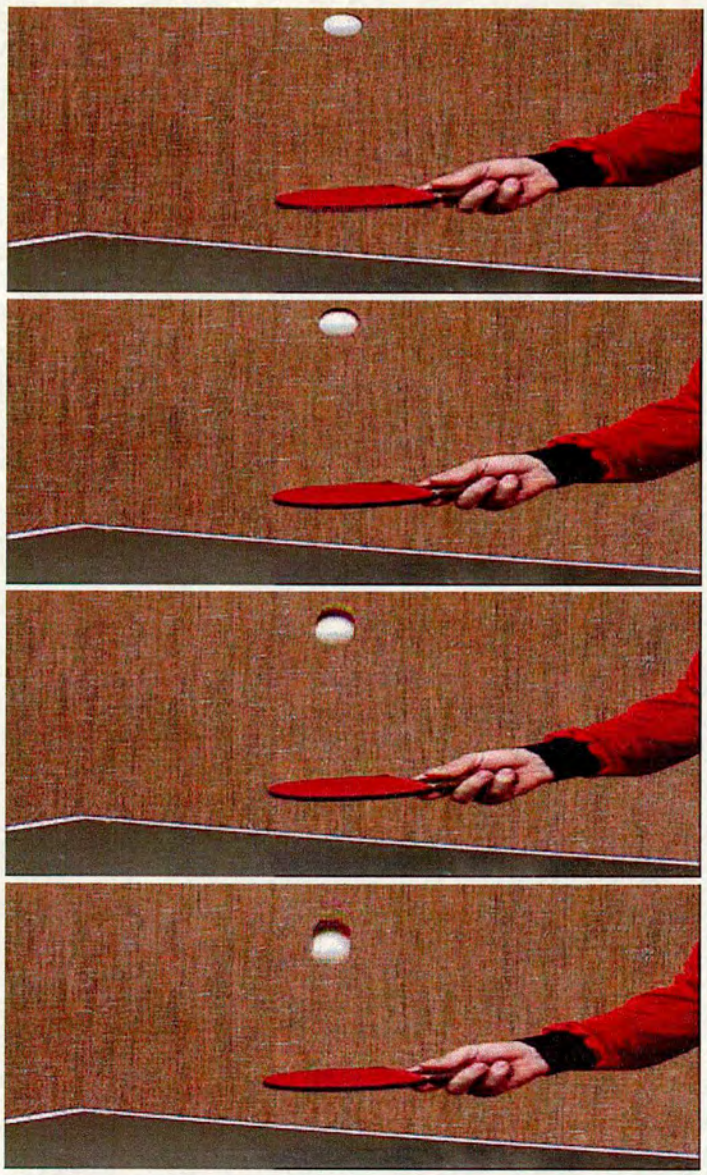

Figure 4.2: *Excerpts from the decoded tennis.m2v sequence (i)*

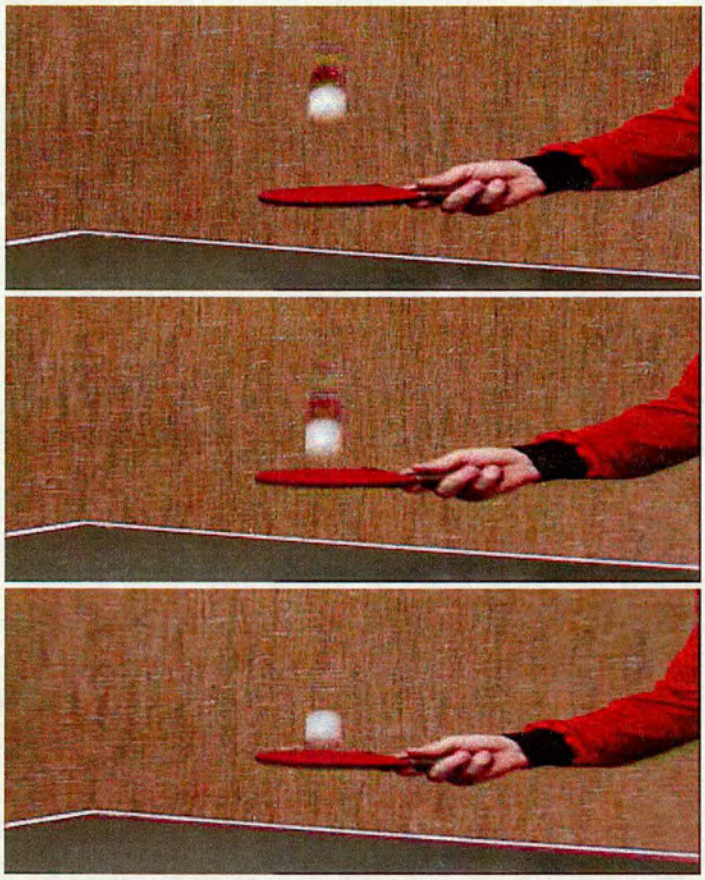

Figure 4.3: *Excerpts from the decoded tennis.m2v sequence (ii)*

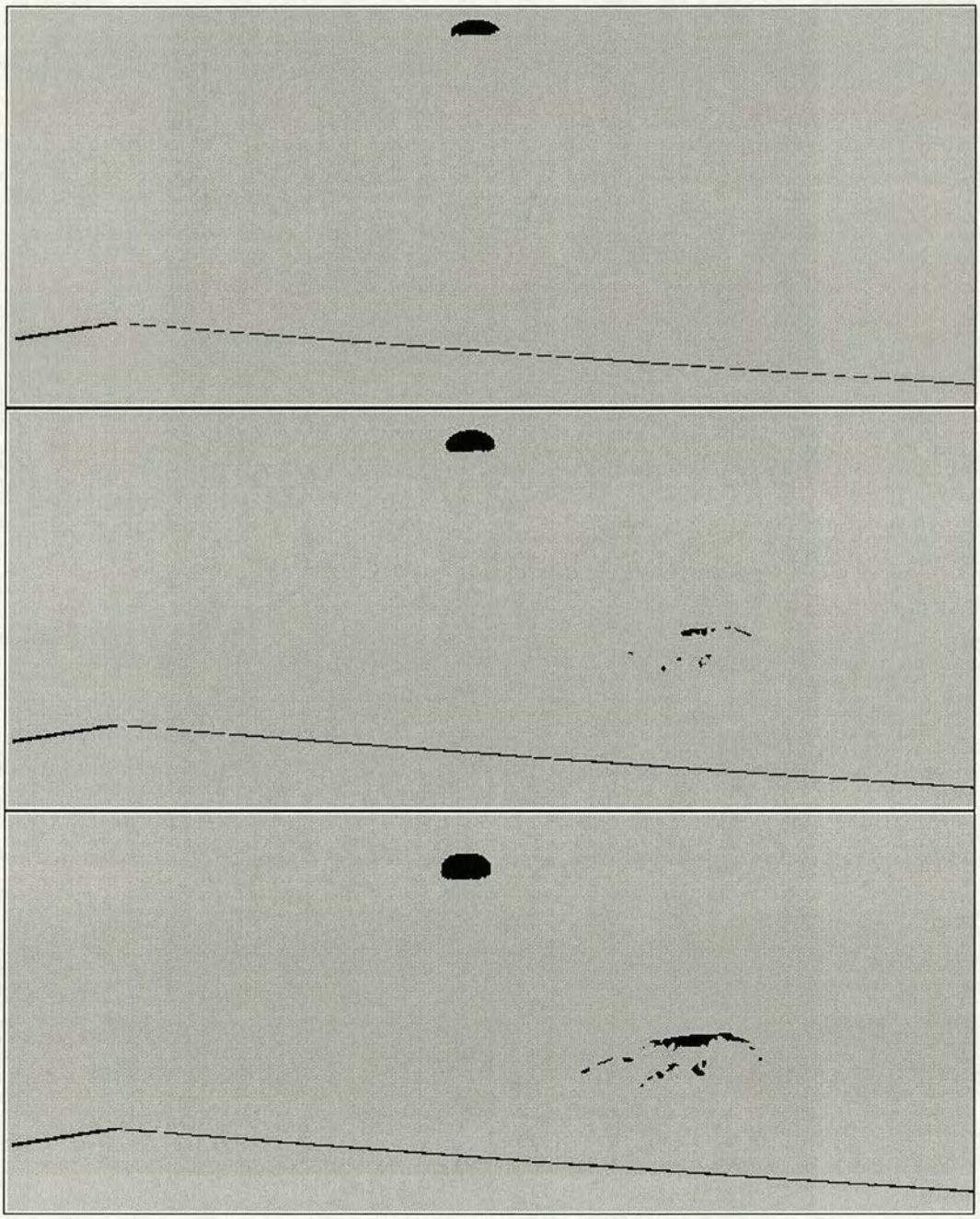

Figure 4.4: *Difference images for the initial test sequence against median background (i)*

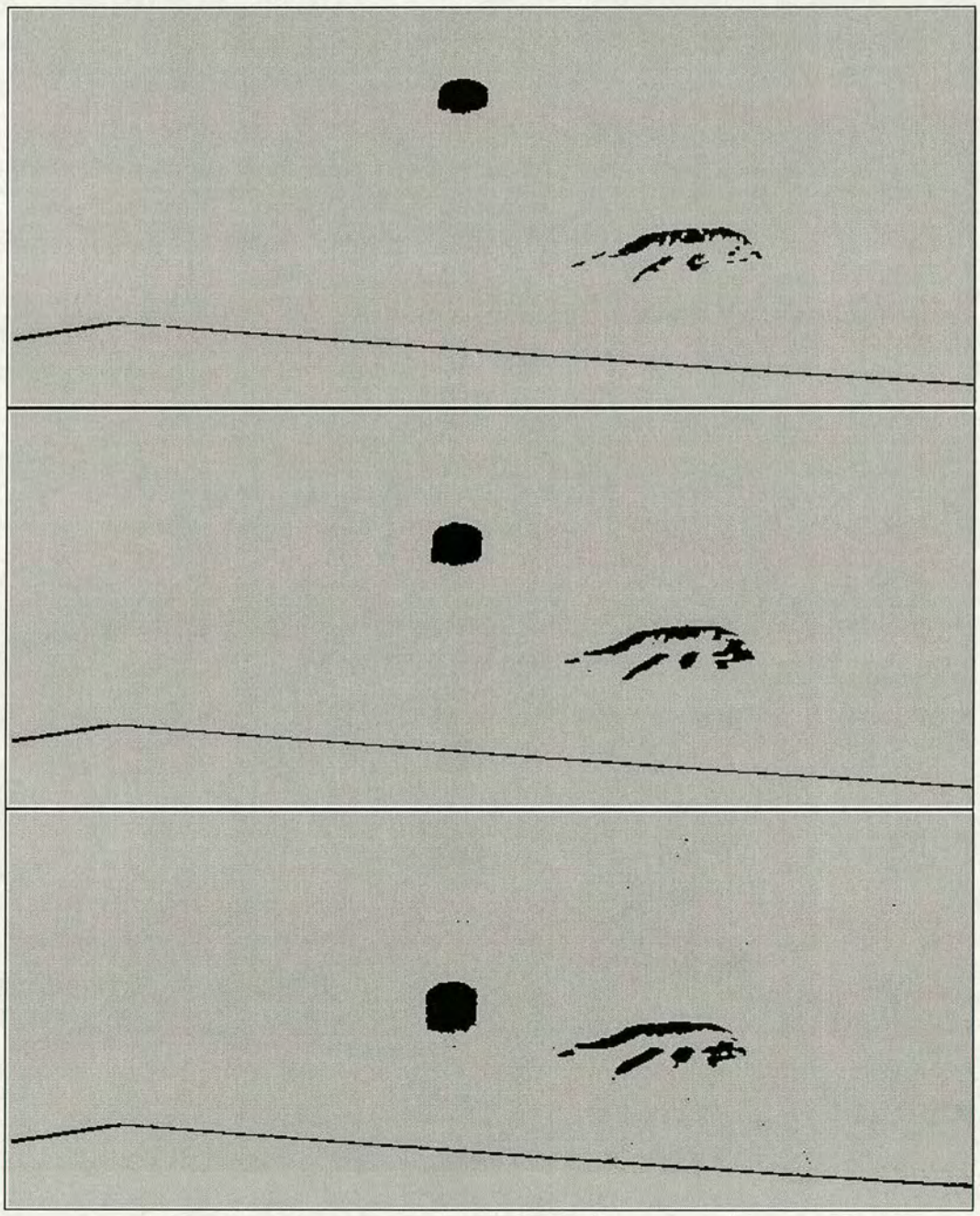

Figure 4.5: *Difference images for the initial test sequence against median background (ii)*

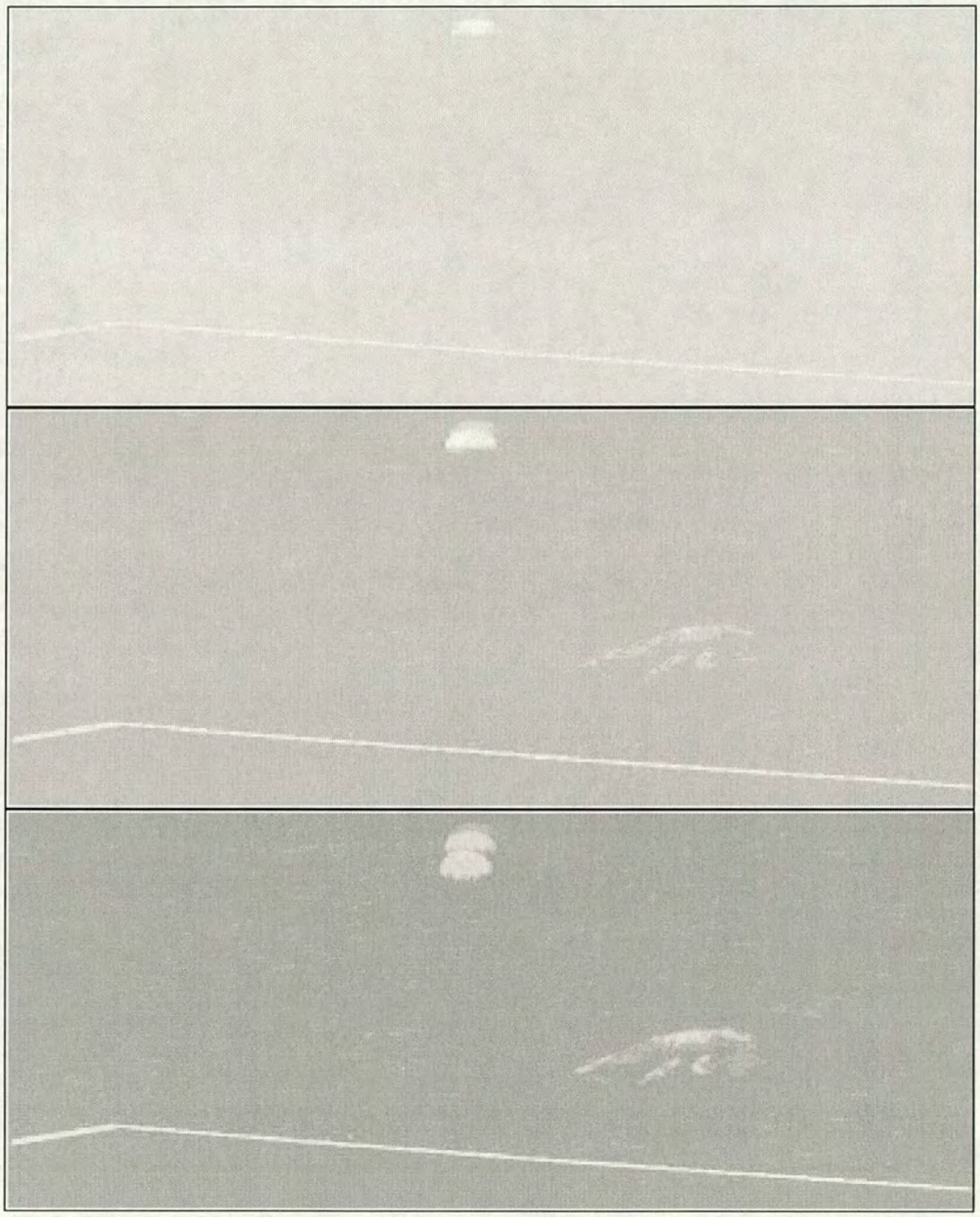

Figure 4.6: *Median background testing for the initial test sequence (i)*

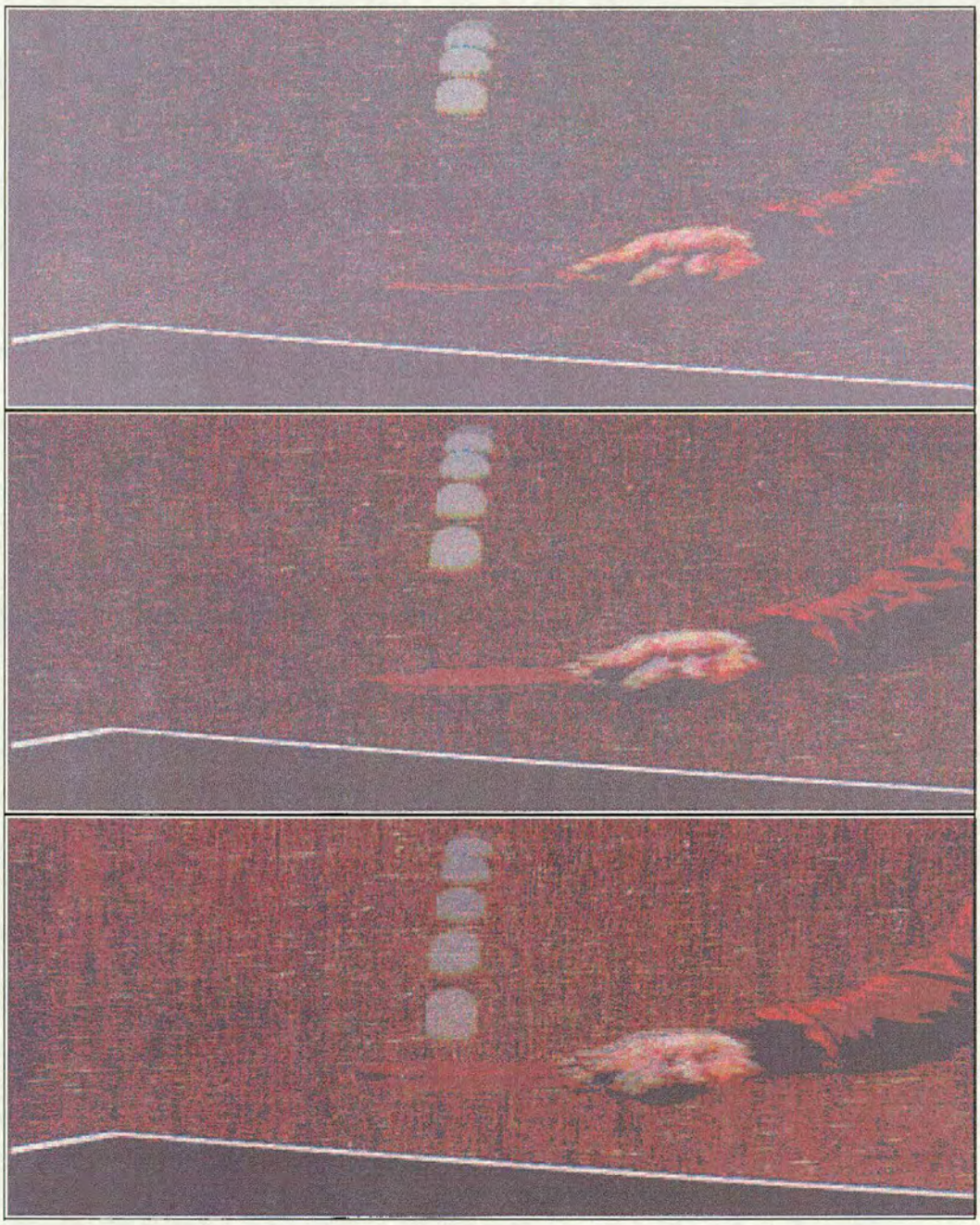

Figure 4.7: *Median background testing for the initial test sequence (ii)*

Firstly, the two visual output routines required are both procedures of the *VS_mix_back* class. Objects of this class contain information in respect of which distributions in each mixture are classified as background, which one in each is matched and whether it itself is background or foreground.

To generate an image-like difference map with values I_c, the procedure is:

For every pixel equivalent position in the *VS_mix_back* object's 2D array
    For all Gaussians in the mixture at the current position
        If the current Gaussian is flagged as matched
            If the current Gaussian is flagged as background, $I_c = 254$
            else $I_c = I_f$

Visual display of the background presents two issues. The first is what value should be chosen as representative of a distribution and can be answered uncontroversially by using the standard approach, giving the mean value. This will not indicate how great a range of values could be recognised as part of that background, but is the best single value option.

Secondly, one of the principal characteristics of this representation is that the background at a single point may not **have** a single representative value at all. To recap, the mixture at any point is a 1D array of *VS_Gaussian* objects, sorted in order of descending probability that each is a component of the background. The number of distributions classified as part of the background at any point is controlled by the variable T and the number of distributions in the mixture.

In the current implementation, the number of distributions classified as background at a point can, in theory, vary from 1 to 3 at any time: an image-like output must be capable of representing this. The chosen solution was to allow multiple image-like entities to constitute the overall representation. The primary background display will present at each point the mean of the distribution most likely to be part of the background; the secondary background display will present at each point the mean of the distribution second most likely and so on. As the mixture is pre-ordered in these terms, the procedure involves simply working through each mixture array in order. To generate a series of three image-like backgrounds with values I_{b_1}, I_{b_2}, I_{b_3}:

For every pixel equivalent position in the *VS_mix_back* object's 2D array
$I_{b_1} = I_{b_2} = I_{b_3} = 254$
bgcounter $= 0$
    For all Gaussians in the mixture at the current position
        If the current Gaussian is flagged as background

Variable	Setting
Median threshold	10
Mix Size (n)	3
Learning rate (α)	0.1
Background fraction (T)	0.5
Distribution initial mean (μ)	300
Distribution initial variance (σ^2)	400

Table 4.1: *Fixed settings for tuning variables*

Increment *bgcounter*
If $bgcounter = 1, I_{b_1} = I_c$
If $bgcounter = 2, I_{b_2} = I_c$
If $bgcounter = 3, I_{b_3} = I_c$

Examples of the background displays for the *tennis.m2v* sequence shown in figures 4.8 and 4.9 for the primary background, figures 4.10 and 4.11 for the secondary background.

When the visual output options were configured, the test sequences were run and the results analysed by eye. The threshold values and other parameters were varied and set at values (table 4.1) which gave an acceptable result, again judged by eye. As noted, the parameters were left open to variation until after running on a longer test sequence, but no changes were found to be required at that point.

4.5 Object growing and labelling

4.5.1 Connectivity

The object labelling process takes as its input the foreground segmentation routine's output, either a trinary difference map (section 4.4.3) or a *VS_mix_back* object (section 4.4.4).

In the first case, there is a sequence of image-like difference maps of the same size as the frames in the source sequence. These trinary images have a distinct value assigned to background, and two further distinct values assigned to object pixels. 'Bright' objects' pixel values exceeded those of the corresponding pixels in the reference image by more than the specified threshold amount, 'Dark' objects' pixel values were below those of the corresponding pixels in the reference image by more than the specified threshold amount.

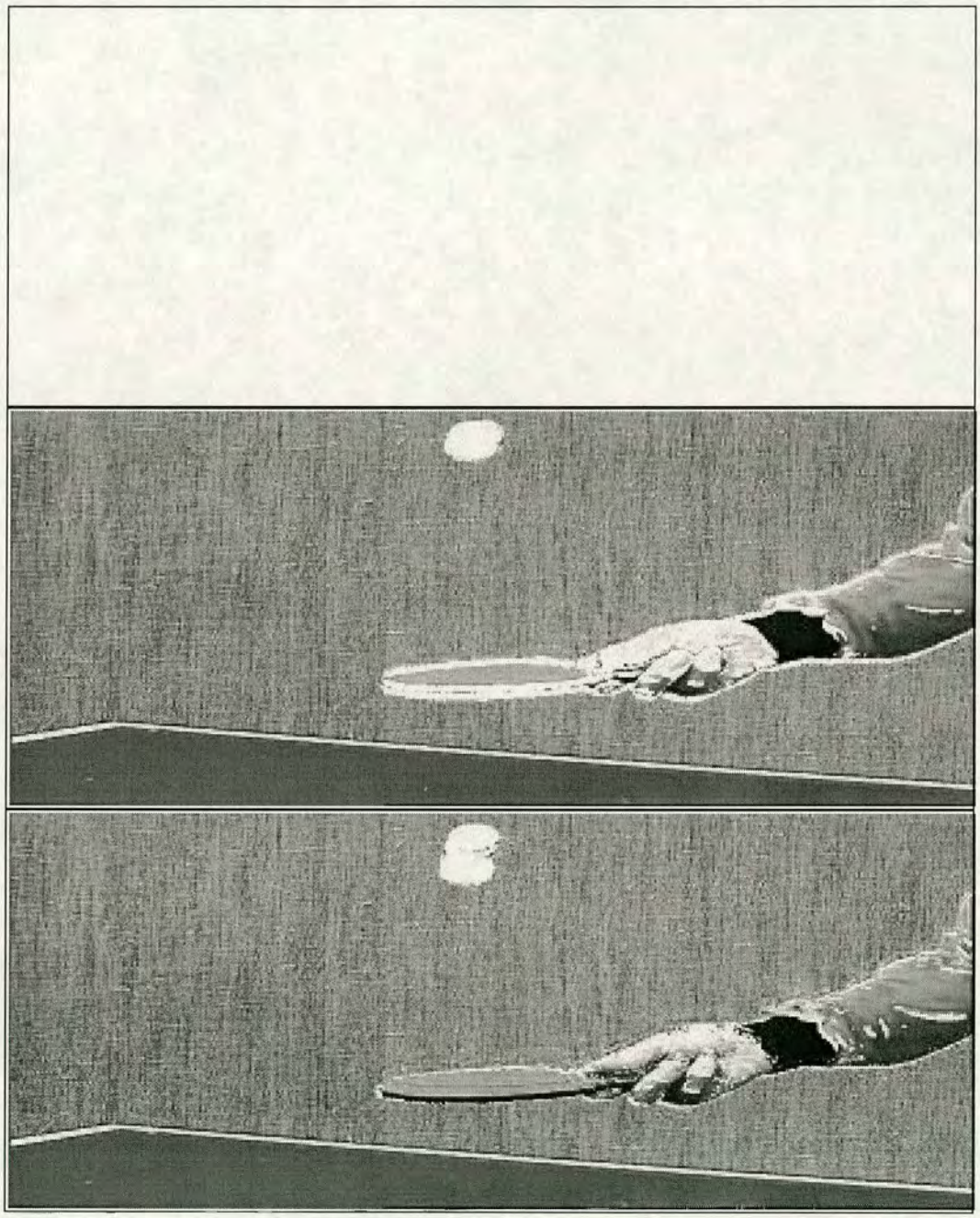

Figure 4.8: *Primary Gaussian background testing for the initial test sequence (i)*

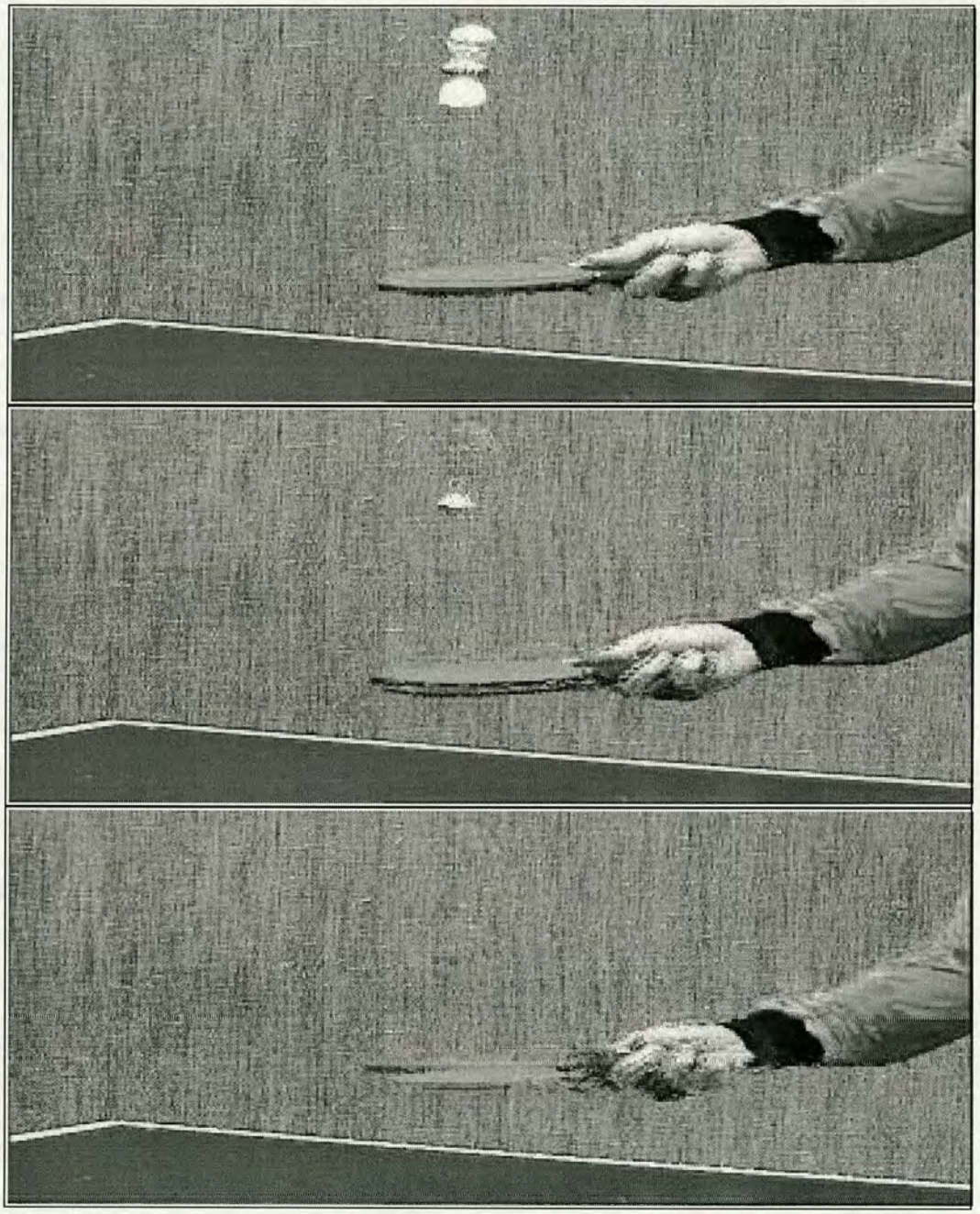

Figure 4.9: *Primary Gaussian background testing for the initial test sequence (ii)*

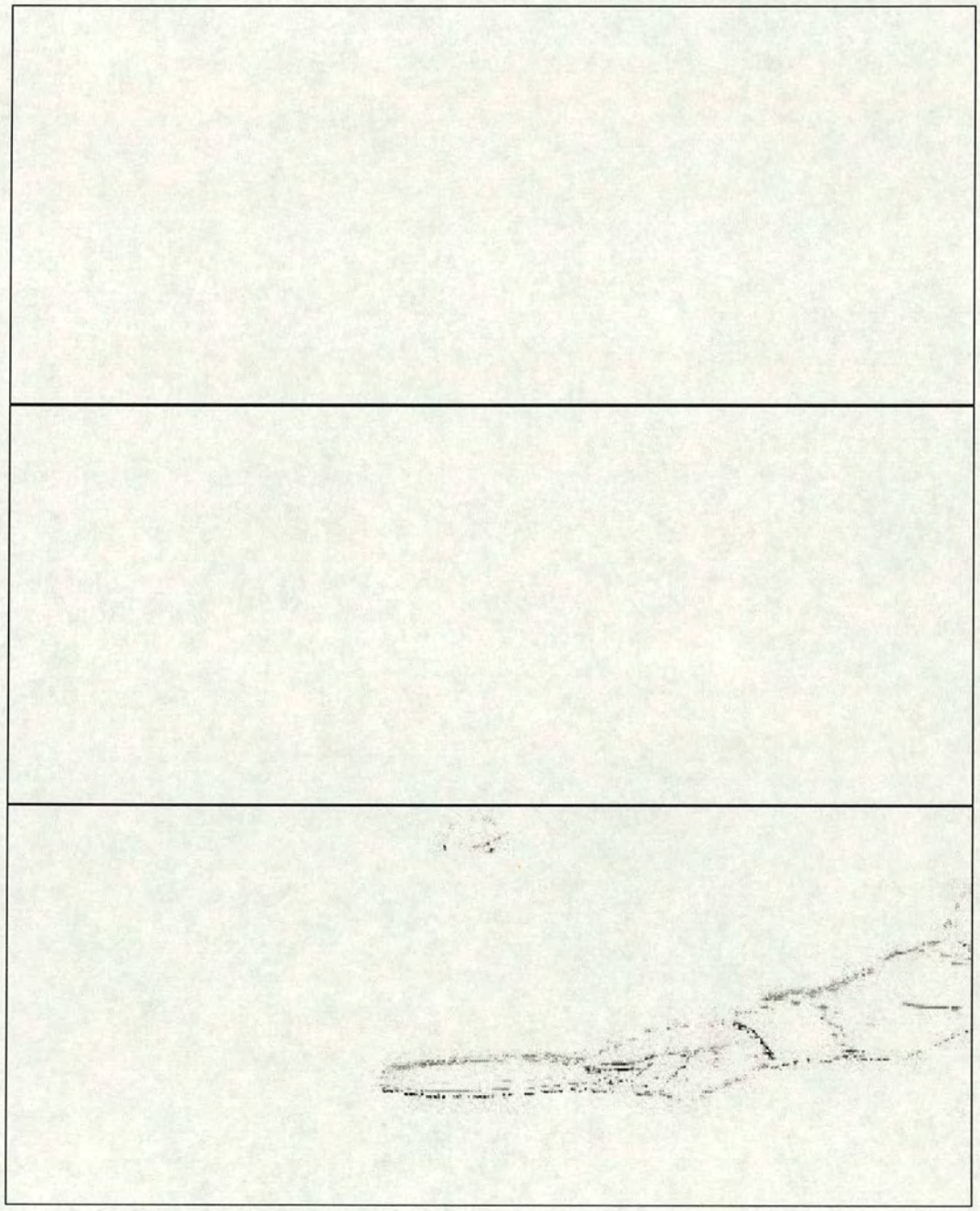

Figure 4.10: *Secondary Gaussian background testing for the initial test sequence (i)*

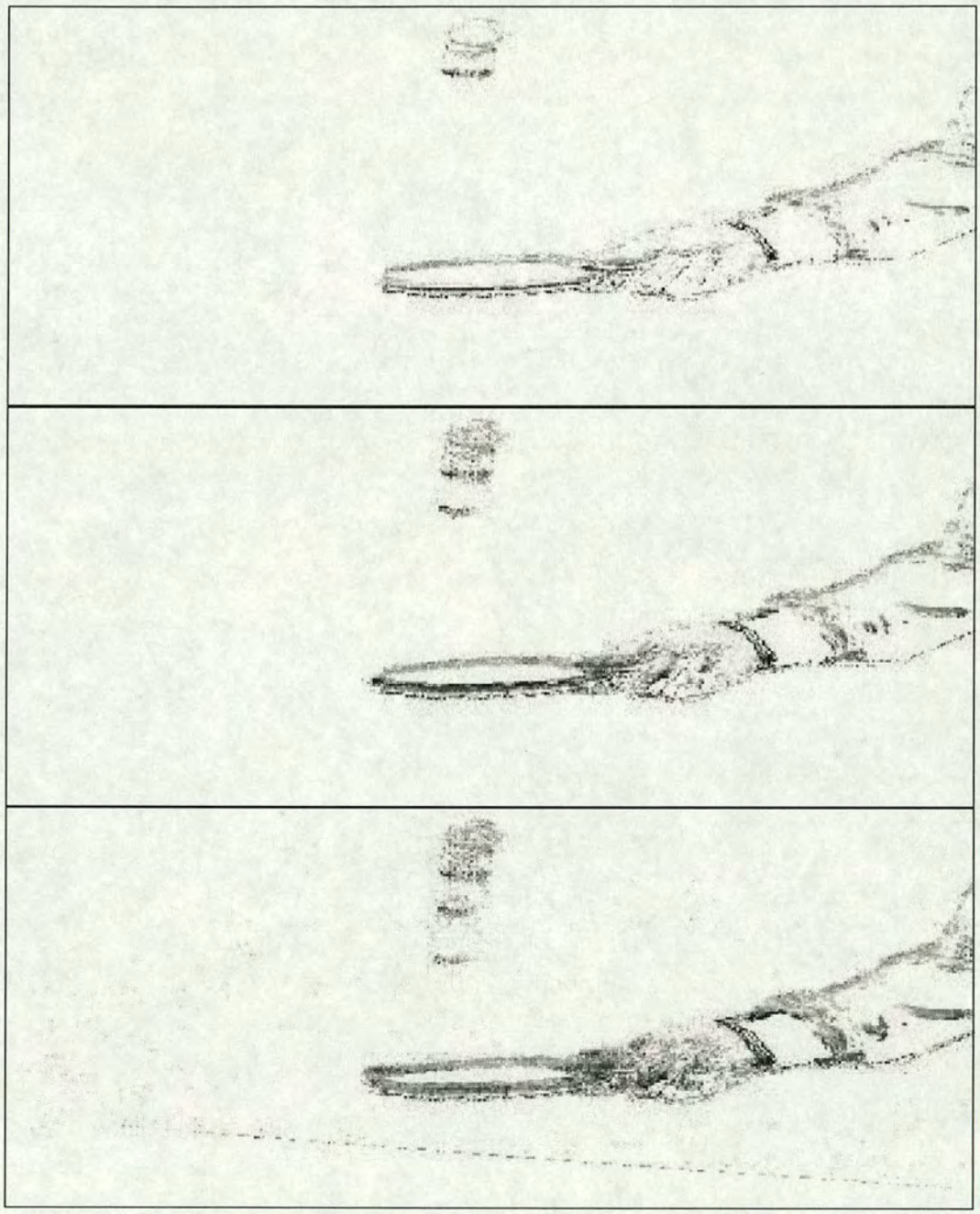

Figure 4.11: *Secondary Gaussian background testing for the initial test sequence (ii)*

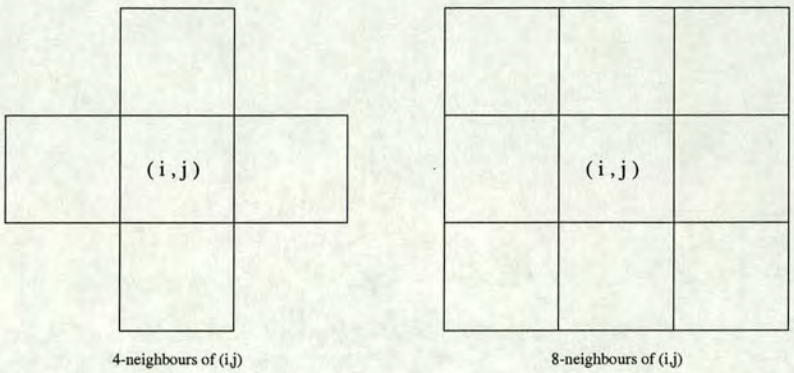

Figure 4.12: *The two alternative neighbourhood areas.*

In the second case, there is a sequence of non image-like objects, each containing a 2D array of *VS_Gauss_mix* objects. The array size corresponds to the size of the frames in the source sequence and each *VS_Gauss_mix* corresponds to a pixel position. Within each mixture, there is one *VS_Gaussian* object flagged as currently matched and greater than or equal to one flagged as (a) component(s) of the background.

Whichever format of input is encountered, the information encoded in the representation is sufficient to specify regions corresponding to background and to foreground. To be able to evaluate the position and properties of elements of the foreground, the input data must be re-represented by re-labelling of the difference map pixels to indicate distinct individual objects. The processes used to achieve this rely on very basic machine vision concepts and processes: analysis of the connectivity of pixels identified as foreground and the assignment of unique labels to all points in the same component.

The connectivity of a pixel is an expression of the number of neighbouring pixels which are in the same *set* as the current pixel. In the current example, members of a set are identifiable by sharing the same pixel value. For our purposes, the sets present in the segmented representation can be considered to be:

- n sets i (for an i range from 1 to the $n + 1$), one corresponding to each of the n objects in the scene
- a general set for background

There are two bases which can be used to define the neighbourhood of a pixel (i, j): we can look at its 4-neighbours $[(i+1), j], [(i-1), j], [i, (j+1)], [i, (j-1)]$ or its 8-neighbours which include in addition $[(i+1), (j+1)], [(i+1), (j-1)], [(i-1), (j+1)], [(i-1), (j-1)]$. From figure 4.12, it can be seen that the descriptions are quite intuitive: when considering 8-neighbours we consider all pixels connected by at least one vertex/corner, when considering 4-neighbours we only consider pixels connected by at least one side.

The choice of which to use depends upon the application. In general, looking at 8-neighbours will require a larger search and will find more 'tenuous' connections; looking at 4-neighbours will require less searching but will not consider diagonal relationships.

In generating the unique object classifications, using 4-connectivity could reduce the occurrence of 'false' inclusion being made of two distinct objects. Certain 'outliers' could be lost (pixels only connected at one vertex) but this should not be a problem, especially as the object is to be approximated by a curve in later stages of the model building process. Indeed restricting ourselves to the more exacting form of connectivity can perform a little extra useful smoothing on the object.

Checking the points in the neighbourhood of a pixel should be implemented as a discrete procedure to allow for re-use as it is a component step in many vision operations. If empirical testing results indicate that objects are being incorrectly split due to 8-connectedness being disregarded, the function could be redefined in terms of finding 8-neighbours. The functional abstraction inherent in correct C++ programming allows for the internal processing to be altered in this manner with no effect on the required methods of using the function, as long as the interface is preserved unchanged.

The desired result of the object growing process is to label the pixels constituting each object with a unique identifier to differentiate the points from those corresponding both to the background and to any other object. Values for the size (in numbers of pixels) and centroid (in frame coordinate) of each object can be calculated simultaneously with the labelling.

The intuitive approaches to the problem (as detailed in [17]) are *connected component algorithms (CCAs)*, where the values of each neighbouring pixel are examined and re-labelled to give labelling of the whole object and exact values for both centroid position and object size. A CCA can be designed most straightforwardly in terms of coding as a *recursive* procedure, and this was the first implemented solution. Excessive resource requirements mandated

the abandonment of this approach entirely and a more complex *sequential* procedure requiring two passes of the image but with less resource demands was substituted as the first candidate solution (section 4.5.2).

The alternative candidate solution implements a boundary following approach, developed from first principles, for each object: some more involved heuristic rules are required in this case to avoid double-counting problems but the results in terms of the objects labelled are identical. The desire to maintain a common interface in terms of input/output characteristics for modules was however only partially fulfilled in this case. The boundary following approach gives an *approximation* to the object size based on the boundary length and assumptions of general shape (see section 4.5.3). This means that results given by the two module alternatives are not compatible and specifically that any succeeding module using size data for comparing objects (e.g. model matching) will malfunction if the objects compared were labelled using different approaches.

The output also varies in that only the boundary is labelled in the second option, whereas the whole object 'blob' is so labelled in the CCA approach so that the visual output differs similarly. This does not prevent comparison between the results of the two approaches and so is not a drawback.

4.5.2 Connected component algorithms

4.5.2.1 Recursive approach

The CCA was first implemented in its most intuitive, recursive version. This involves the methodical review of the values of neighbouring pixels of a foreground object pixel.

The object classifying process has the following steps:

Build an object map with a *VS_image* of size equal to the original frames
Initialise variables *size*, *x_accumulator*, *y_accumulator*, *object_counter* to zero
For every pixel equivalent position in the segmented representation
    If the corresponding value indicates an unlabelled foreground point, we have a new object
        Increment *object_counter*
        Set corresponding object map pixel to value of *object_counter*
        Relabel the original to indicate that this is no longer an unlabelled foreground point
        Increment size; add x and y values of current pixel to *x_accumulator* and *y_accumulator* respectively
        Recursively check neighbouring pixels, and label/accumulate until no unlabelled foreground neighbours are located
    If the final value of *size* is greater than or equal to the threshold value
        Store *size* value in a 1D *size[]* array at a position indexed by the value of *object_counter*

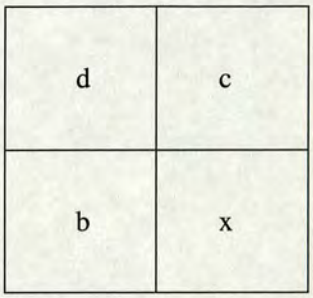

Figure 4.13: *The neighbourhood area of pixel x used in the sequential approach, first pass*

Calculate centroid points as $c_x = \frac{x\_accumulator}{size}$, $c_y = \frac{y\_accumulator}{size}$
Store c_x, c_y in a 2D *centroids[]* array at a positions indexed by the value of *object_counter*
else
Reset object map pixels with a value of *object_counter* to background
Decrement *object_counter*
Reset *size*, *x_accumulator*, *y_accumulator* to zero

The output of the module is a *VS_object_map* object, the principal data attribute of which is a *VS_image* which is a fully labelled representation of the individual objects in the input. This image-like component is supported by two arrays indexed by unique object label (so having unit offset, i.e indexed from 1) indicating the object size in pixels and the centroid position.

As noted above, although intuitive, the recursive process is very demanding in terms of computer time resources on a serial system. The time taken in processing images during initial testing of approximately 15 minutes for the simple eight frame *tennis.m2v* sequence was prohibitive and so the recursive approach was replaced by a two-pass sequential implementation.

4.5.2.2 Sequential approach

This approach requires two passes of the segmented representation: in the first only three neighbours (figure 4.13) are considered to give an initial labelling and construct an equivalence table. The second pass is used to relabel the object map with the lowest member of each equivalence set as recorded in the table:

Build an object map with a *VS_image* of size equal to the original frames
Initialise variables *size*, *x_accumulator*, *y_accumulator*, *object_counter* to zero
For every pixel equivalent position in the segmented representation
If the value indicates an unlabelled foreground point

If neighbour d is labelled, set this value at pixel x and set entry x in the equivalence table to d.
else if neither c nor b are labelled, we have a new object.
Increment *object_counter*
Set this value at pixel x and similarly set entry x in the equivalence table.
else if only b **or** c are labelled or b **and** c are labelled and $(b = c)$
Set this value at pixel x and similarly set entry x in the equivalence table.
else if b **and** c are labelled, and $(b \neq c)$
Set the value of c at pixels x and b and similarly set entries x and b in the equivalence table.
For all entries in the equivalence table beginning from 1 upwards
Replace each entry with the lowest member in its equivalence set
Accumulate the total number of entries in each equivalence set
Review entry counts for each equivalence set
If total is below the specified threshold, zero all entries and relabel the table accordingly.
else store the total entry count for the set
For every pixel equivalent position in the segmented representation
If the value corresponds to an equivalence table entry, replace the label with the lowest equivalent set value
Accumulate and store (x,y) values to calculate centroid.

Using this sequential approach resulted in frame processing times improved by a factor of 10, taking approximately 90 seconds for the test sequence.

4.5.2.3 Implementation points

Trinary difference map values

In processing the trinary object map, relabelling to indicate that a point is no longer unlabelled foreground is achieved by resetting the value to that of *object_counter*. For this to work, it is essential that none of the background, bright object or dark object pixel values should be in the same range as possible unique object identifiers. This was ensured by using positive integers from 1 for unique object identifiers, zero for background and negative values for bright objects and dark objects.

Object map attributes

Only objects which exceed the specified threshold size are included in the *VS_object_map*: this constitutes a supplementary noise reduction process where any detected object below threshold is **assumed** to be not significant. In the current implementation, the value was empirically set at 1000 pixels. Clearly using an absolute value will not be suitable in all applications as the significance of a size in simple pixel number terms will vary with the scene magnification and image size.

With respect to scene magnification, inherent limits on the abilities of the system are where pedestrian objects are so small as to be indistinguishable from scene noise at one extreme and so large as to not be encompassed within the frame at the other. In respect of the significance of image size (in numbers of pixels) the threshold could be set to vary in proportion to changes made from the default expected frame size. The value of threshold chosen for a particular scene magnification/image size combination must reflect a balance between noise reduction and system ability for distant pedestrians.

The size of an object was recorded by accumulating the number of pixels labelled during the process in the array as above. Where this value was below a specified threshold, the object pixels were reset to the background value. Similarly, the x, y coordinates of each identified point were stored during labelling: division of the accumulated totals by the number of points labelled gives the object's centroid position.

The two required supplementary arrays were constructed at a size corresponding to the maximum theoretical number of discrete objects which could be detected (O_{max}) at the specified threshold level, calculated by:

$$O_{max} = \frac{image - area}{threshold - size} \tag{4.28}$$

I/O functionality

As the object growing process constitutes a 'bottleneck' in many vision processes, the current one included, functionality was included to allow for file storage and loading of *VS_object_map* objects. This was implemented by using the extant *VS_image* i/o function for storage of the image-like component as a *.ppm* file and to augment this by filing the non-image data in a supplementary *.Omap* file. A constructor for the *VS_object_map* class was added to allow objects of this type to be built using such a pair of files.

This facilitated the efficient testing of subsequent modules, using a sequence of file pairs as input, circumventing the need for object growing on each run.

Boundary conditions (sequential approach only)

Special cases of the initial labelling procedure were required for pixels along the top row and left column. These simply substituted abbreviated searches which assumed the out-frame positions to be background.

Equivalence table (sequential approach only)

The equivalence table was first implemented as a 2D array of size ([2][O_{max}]) where O_{max} was given by equation 4.28. Further consideration revealed that a much larger array size may be needed, with the limiting case in the initial labelling of a new object every four pixels. To avoid possible overflow, O_{max} must be restated here as:

$$O_{max_2} = \frac{image - area}{4} \tag{4.29}$$

The maximum number of objects here corresponds to the hypothetical case where the frame is 'full' of single-pixel sized objects. Referring to diagram 4.13, this situation requires one object every four pixels, corresponding to the formula given.

The first element at each object label index is used to store the current equivalent value for that label, initialised at itself. The second element is for storage of the current estimate for the size of the equivalence set. After the initial setting of equivalence values in the first pass over the segmented representation, the table is relabelled as follows:

For each non-zero entry in the equivalence table
- Set v to the entry value, i to the entry index
- If $v = i$ (Base Case: no lower equivalent)
  - Add element size to current cumulative size
  - Replace element size with current cumulative size
  - Pass value out as lowest equivalent
- else (Recursive case: get values from base case)
  - Add element size to current cumulative size
  - **Recursive step:** repeat process at lower table entry using v as the new index
  - Reset v to lowest equivalent passed up from Base case of recursion
  - Replace element size with current cumulative size

4.5.3 Boundary following object labelling

4.5.3.1 Boundary tracing

To extract the boundary pixels for each object and store them in an array requires a two pass connectivity-based procedure. The approach was developed from first principles based on consideration of the geometrical properties of the scene and is more involved than the intuitive CCA method.

The basic principle is conceptually straightforward: instead of examining all neighbours throughout the body of each object to assess connectivity, the boundary pixels are traversed and labelled. These are all that is required to calculate the accurate centroid and principal axis of the object and form a suitable basis for estimating its size. The actual process is only this simple in the trivial case of a single object in scene: more complicated scenes require the application of a set of heuristic rules and approximations.

Build an object map with a *VS_image* of size equal to the original frames
Initialise variables *size*, *x_accumulator*, *y_accumulator*, *object_counter* to zero
For every pixel equivalent position in the segmented representation
    If the value indicates an unlabelled foreground point, check its neighbours
        If any are labelled already, set *oldflag* to 1
        If any are background, set *edgeflag* to 1
    If edgeflag and **not** oldflag, we have a new object
        Increment *object_counter*
        Relabel original to indicate this is no longer an unlabelled foreground point
        Increment size; add x and y values of current pixel to *x_accumulator* and *y_accumulator* respectively
        If x and/or y coordinate constitutes maximum or minimum seen so far, record the values.
        Record this point as *start* and *current*
        Check clockwise round neighbours from b in figure 4.13 for the first unlabelled foreground point.
            Set that point as *current*
        Repeat for each new boundary point until *current* = *start*
    Calculate a size estimate using x and y maximum/minimum values and the approximation that the object is a vertical ellipse.
        If size is greater than or equal to threshold:
            Store *size* value in a 1D *size[]* array at a position indexed by the value of *object_counter*
            Calculate centroid points as $c_x = \frac{x\_accumulator}{size}$, $c_y = \frac{y\_accumulator}{size}$
            Store c_x, c_y in a 2D *centroids[]* array at a positions indexed by the value of *object_counter*
        else
            Reset object map pixels with a value of *object_counter* to background
            Decrement *object_counter*
    Reset *size*, *x_accumulator*, *y_accumulator* to zero

Using this approach, when an object has been labelled and accepted as above threshold size, it retains a 'protective layer' which prevents it being recognised as a new object and double counted. Conversely, objects below threshold size are 'winnowed away' a layer at a time: this involves some extra processing effort until the object fully dissolves, but as the objects are by definition small, the additional processing incurred is not large.

Compared with the output of the CCA approaches the most obvious divergence is that the image-like component is labelled only at the object boundaries, but the object position (and later, principal axis) data will be unchanged. The other major change is in a size estimate

being used for each object. This applies the clearly unrealistic assumption that all objects are vertically oriented ellipsoids, but the results are quite adequate for comparison of relative object sizes.

4.5.3.2 Implementation points

Boundary conditions

Special cases were required for pixels along the edge rows and column and further different cases at the corners. These simply implement that the out-frame positions being ignored in the clockwise neighbourhood search.

Single point objects

When a single foreground pixel is located, the clockwise neighbourhood search to find the next boundary pixel can cause an infinite loop. The addition of a neighbour review counter and a check that this has not exceeded eight provides the solution to this. If the counter does exceed eight, the single foreground pixel is set to background as noise.

Size estimation

The size calculation has two principal purposes: the elimination of objects below a specified threshold value and a metric for scaling valid objects.

In the first respect, the precision and indeed accuracy of the evaluation is not critical as long as the estimate is conservative in respect of pedestrian type shapes. The vertical ellipsoid calculation is based upon the recorded maxima and minima for both x and y coordinates and will err toward overstating size. Any objects erroneously accepted will be eliminated if they fail to be classified as pedestrians, at the cost only of some extra processing in the matching stage. The use of an estimate which assumes in advance a pedestrian-type shape is quite suitable.

For the second purpose, the size measurement may be used to eliminate differences in scale when comparing two objects (for example a model and a new object). They key point here is that the use of an approximation will not cause problems as long as it is applied consistently. If this object labelling approach is used, for example, in building the model of a pedestrian, it must be used in matching models in each new scene and vice versa. This is not a problem as the module choices will be fixed in the final implementation and so a mis-match of labelling approaches cannot occur.

4.5.4 Functionality testing

The result of the object labelling process is a sequence of objects, each with:

- Image-like object maps of the same size as the frames in the training sequence, with background pixels at zero and object pixels labelled with unique identifying integer values from 1 to $n + 1$ (where $n =$ the number of objects).
- An array of object sizes ordered by object number.
- An array of object centroids ordered by object number.

The exact format of labelling and the size estimate varies dependent on the candidate solution employed.

To evaluate these results, it is again necessary for inclusion of procedures for the image-like output of intermediate results within the implemented code, which outputs will be in the *VS_Image* format to allow reuse of the i/o options of the class.

In both cases, the procedure uses the object labels as a base for assigning pixel values:

For every pixel position in the object map *VS_image*
Get the current pixel value (O_c)
If $O_c = bg\_value, O_c = 254$
If $O_c = O\_number, O_c = S \times O\_number$
(If $O_c = fg\_value, O_c = 254$)

S is an integer multiplier for numbered objects used only for increasing contrast to enhance the ease of reading of the visual output by eye and was set at 20. The third case, in brackets, is relevant only for the boundary following object growing method and removes the 'filling' of the object boundaries.

Evaluation is by visual comparison of the results with the difference maps from the same source data.

4.6 Object classification

4.6.1 Approaches to classification in vision

Of the areas of vision research examined as distinct modules for the construction of the overall application object classification is the one associated with the largest volume of previous work. The range of approaches investigated is correspondingly wide (see Chapter 2) and the potential for interesting and useful comparisons between alternative methods extensive.

Given this context, several broad considerations were used to guide the choice of the approaches to use as candidate solutions, in addition to the general points given in section 4.2:

- At least one candidate solution will be based upon an approach drawn from the existing literature. This will be one that has strong qualitative results for performance within a surveillance setting.
- The methods implemented should reflect distinct subclasses of approach to object classification (the chosen central comparison was *model-based* and *pixel-based* - see below)
- Following from one of the general points, it **is** possible to find methods within this area of very widely varying complexities. The two methods chosen should then represent distinctly different complexity levels.
- The candidate solution based upon an approach drawn from the existing literature should be one which has not previously been adapted to work on a distributed processing platform with limited on-camera processing. The implementation of a variant suitable for such a system will then be of further research interest in itself.

Two distinct subclasses of approach to object classification which provide an interesting opportunity for investigation were *model-based* and *pixel-based* methods. *Pixel-based* methods to image analysis rely on analysis of the pixel values within an image compared with the values of neighbouring pixels, of 'connected' pixels (see section 4.5) and of pixels globally within the image. Heuristic rules can be used to evaluate the results of this analysis which can range from simple segmentation to a series of complex mathematical operations.

All the methods detailed for background estimation, foreground segmentation and object labelling are *pixel-based*, as are most basic vision processing operations. A key feature which

characterises this subdivision of approaches is that such approaches **do not** involve the use of a pre-prepared model appropriate to the process. This does not refer to the use of a mathematical model of a process, which is implicit in all pixel based approaches, but rather a model of a particular object or feature which is to be used in the processing.

This definition is not completely infallible: where, for example, corner detection or edge detection uses a simple pattern as a template for locating significant pixels (e.g. a line in one dimension represented by the pattern $0 - 1 - 0$), this is usually classified as *pixel-based* rather than *model-based*. The key is the complexity of the model and its construction: in the pattern example, the features are called *primitives*, corresponding to basic building blocks of image-like representations which are usually defined by hand. In the *pixel-based* method proposed below, the assumption that a pedestrian can be classified by shape is a simple 'model' of this type.

To constitute a *model-based* approach, the models are generally complex, specific to a particular type of object and most often *learned* in some pre-processing stage. The model can be of widely varied entities including a specific object (2D or 3D), a view of an object, a type of object, an object's variation over time, a trajectory or a pattern of behaviour.

Whereas, in *pixel-based* approaches, processing is conducted on an image directly, with no specific need for supplementary processing in advance, the first step in a *model-based* approach must be construction of one or more appropriate models. The models may be constructed 'by hand' but most often are learned from representative examples of the entity which the model is to represent. The models are then stored for use in the main image processing session using some form of model matching.

The model is compared with an hypothesised instance of the entity which it represents in the image and some metric is used to compare the two. The result of this matching process can be an object classification (as in the current example) or a result such as identification of an individual or analysis of a trajectory or behavioural pattern.

As can be seen from the examples discussed in Chapter 2, research endeavour in the field of classification has primarily concentrated on a wide variety of *model-based* approaches. The models applied to represent the human body (a critical task in pedestrian detection) take different approaches to capturing the intrinsic flexibility and thus variation in pedestrian shapes.

With *stick figure* models, the body is represented by a skeleton of line segments, generally connected at their end points. *Volumetric models* attempt a better representation of the body's three dimensional complexity by replacing the line segments with generalised cylinders which allow better recognition over multiple viewpoints.

Straddling the two, *cardboard models* have been used [76] with simple two dimensional shapes as segments to give a balance between performance and computational requirements. The work on *flexible 2D models* of a body as a spatiotemporal entity, discussed in more detail below, represent a body using an average shape and a set of characteristic distortion parameters.

Rigid models, whether 2D or 3D have limited use in modelling pedestrians, and are more frequently used for tracking inherently rigid objects, such as cars.

The relative merits of *model-based* over *pixel-based* approaches can be stated qualitatively as follows:

- It is difficult to achieve object specificity and flexibility in object range when using *pixel-based* approaches for detecting complex objects. It it not possible to encode complex characteristics of an object for recognition without using a model. These approaches are restricted to using more general characteristics such as colour range, size, position, shape, pose, position and trajectory.
- *Model-based* approaches require the model building preprocessing step. For this, knowledge of sufficient details of the element to be matched must be known and both representative data and sufficient time must be available for training.
- The model building process must be carefully performed if it is to capture the appropriate characteristic features of an element. An example of where this can fail in the first respect is an anecdotal account of a 'tank detector' trained on a set of images both containing tanks and without them. The detector seemed to operate well, but wider analysis of the performance showed poor results: all scenes containing tanks had been taken at night and the actual result was a 'night detector'.
- *Model-based* approaches have the advantage of being precisely 'tuned' for a specific target element and so are less susceptible to false identifications. The downside of this is that the models can be over-specific and unable to match with the general case of an object. They can also be confounded by augmentation of an object (for a pedestrian this

Approach	**Pixel based**	**Model based**
Basic element	Pixel cluster	Pedestrian model
Fitness Measure	Shape, orientation	Deformation parameters
Occlusion Handling	Poor	Good
Pedestrian Differentiation	Bad	Poor
'Augmented' Pedestrians	Poor	Bad
Distant objects	Poor	Bad
False alarm rate	High	Low
View dependency	High	High
Scale Dependency	Low	Low
Computational requirements	Low	High
Memory requirements	Low	High

Table 4.2: *Pixel and model based object classification comparison*

may be addition of a hat, bag, umbrella or pram).

- The handling of occlusion (where only part of an element is observed) is possible in *model-based* approaches, whereas it generally presents a greater difficulty to *pixel-based* approaches.

- It is possible to use *model-based* approaches to make *predictions* about a full element from a partial representation (this can be of future development if the model includes a temporal component).

Despite this wide range of qualitative information on the relative merits of *pixel-based* and *model-based* based approaches, there has been no quantitative performance analysis conducted in this area. *Pixel-based* approaches have been somewhat neglected, based largely on subjective evaluations as summarised in table 4.2. The characterisation of the performance of the current system overall makes some contribution to providing a comparison of the results of two such approaches on a common data set. The comparison is restricted to being between two specific methods and moreover two specific implementations, which is a key limiting factor in its generality.

4.6.2 A model-based object classification approach

With the majority of object classification approaches in the literature being model-based, the number of choices for the candidate solution is extremely large. The criteria for the specific choice are outlined in section 4.2 and augmented in section 4.6.

There are several major subdivisions into which the model-based approaches developed may be gathered as introduced in section 4.6.1.

- Rigid models (2D & 3D)
- Articulated models (1D, 2D & 3D segments)
- Flexible 2D models
- Spatiotemporal models

The selection from these subdivisions of a modelling approach to develop and modify for use within the current system was principally motivated by an analysis of the extant qualitative results from previous research endeavour in the area (see Chapter 2).

Rigid models are, as previously noted, most useful for tracking cars and similar rigid objects. *Articulated models*, although compact, require identification of individual components of a pedestrian body. In the anticipated application, surveillance cameras may be positioned at an oblique angle, looking down on the scene, the foreshortening effect this causes potentially presenting severe problems for this task. Accordingly, these models were considered a suboptimal choice, although the work in [76], [75] suggests the problem may be tractable.

Generally, a 3D model of a 3D object presents the potential to be most robust over angle, but it is desirable that the model building be feasible using a single camera and its distributed processing unit. Although a general pedestrian model is conceived as being provided with the application, for greater versatility the option of building a view-specific model should be available. Further, the resource requirement of building a *flexible 3D model*, even off-line, that captures the variations likely in a pedestrian are considered prohibitive.

The *flexible 2D model* approach has been presented [25] applied in the very situation of interest in terms of camera positioning (remote, overhead/oblique) and scene type (pedestrian thoroughfare). The approach is essentially pose invariant as it portrays the nature of character-

istic perturbations at fractional increments along the boundary of a scaled mean shape. With foreshortened objects, the increments should simply 'slide' along the boundary in response.

Further, the approach has been shown [105] to be amenable to the construction of *Spatiotemporal* models of object behaviour. These would be of interest in extensions of the system to predict and/or evaluate pedestrian behaviours.

The specific approach investigated was suggested by the Integrated Traffic and Pedestrian Model-Based Vision System [25], product of a collaborative effort between Computer Vision Groups at Leeds and Reading Universities. First, a broad classification of objects in the scene is made as either flexible or rigid bodies, the focus in Leeds being to represent the trajectory of a flexible object (e.g. a person) in a scene geometrically, using representative landmark points and including information on interaction with mapped stationary objects (e.g. cars).

In the approach, a model can be constructed using a representative sequence of a single pedestrian in motion. In each frame, the boundary of the pedestrian is extracted and approximated mathematically as a *cubic B-spline*. This mathematical representation allows a model of the characteristic outline variations to be constructed using *principal components analysis (PCA)* to generate the corresponding *eigenvalues* and *eigenvectors*. The model consists of a scalar size measurement, a centroid offset vector and arrays containing the average shape boundary, the eigenvalues and the eigenvectors.

To classify a new object, the average shape boundary is scaled and placed over the object (using the centroid offset). A dynamic simultaneous minimisation process is employed to find the combination of eigenvector-generated distortions required to minimise the difference between corresponding model and object points. For an object to be classified as a pedestrian, both the minimum distance and the distortion required must be below specified threshold values.

This approach fulfils the criterion previously specified (section 4.6) in that the original implementation gives good qualitative results, involves a high level of mathematical complexity and has not previously been adapted to work on a distributed processing platform with limited on-camera processing. The approach has been previously implemented for a system with fixed, stationary cameras observing pedestrian surveillance scenes [25] and so is *prima facie* appropriate for the intended application

To implement a variant of this approach, the individual steps from an input in the form of a

labelled object map (section 4.5) are:

1. Boundary extraction
2. Boundary reordering
3. Spline fitting
4. PCA: Principal components analysis (for the model building)
5. Model matching
6. Registering and listing pedestrian objects

Steps 1 to 3 are used in in both the model building and model matching processes, steps 4 and 5 are used in model building only and the balance are used only in model matching.

4.6.2.1 Boundary extraction

Whether the CCA or the boundary following object labelling module is used, the input data will be a sequence of image-like object maps of the same size as the frames in the source sequence. In either case, the first task is the extraction and storage of the boundary points of each object. Each individual object's boundary is simply traced to give a list of points using an algorithmic approach very similar to that used in the boundary following object labelling method in section 4.5.3.

Allocate memory to store the boundary points for each object at maximum size
Initialise *size* variable
For every pixel position in the object map
    If the value indicates an labelled foreground point not yet marked as traced, we have a new object.
        Record this point as *start* and *current*
        Put *current* into first space in a new boundary array
        Increment size
        Check clockwise round neighbours from b in figure 4.13. for the first unlabelled foreground point.
            Set that point as *current* and record as the next point in the boundary array.
        Repeat at for each new boundary point until *current* = *start*
    Mark this object/label as traced
    Record *size* in *boundary_sizes* array
    Reinitialise *size*

The procedure is a simpler than the extraction of boundaries from an unlabelled segmentation representation as a simple recording of which labelled objects have been traced replaces the heuristic approach needed in the absence of labels.

Implementation points

The boundaries are stored in a 2D *boundary_array* of size *[n_objects][max_size]*. The evaluation of *max_size* will vary depending on the previous module choice. If the boundary following approach was used, the size of the target boundary is already known and can be used immediately. If the CCA approach was used, *max_size* must initially be set to the known value, that of the object *blob* size in pixels, the limiting case of the boundary length.

To achieve a more efficient memory usage, the array can be 'cropped' after all of the objects have been traced. The boundary data must be copied into a temporary array while the *boundary_array* is deleted and reconstructed at the reduced size.

4.6.2.2 Boundary Reordering

For each object, we now have a list of points corresponding to the boundary. To build a general object model over multiple frames the variations of corresponding points on the boundary over time must be evaluated. The points compared need to be arranged in a consistent order and so some criterion must be specified to choose where an ordered list of points starts for each new observed shape. The boundary extraction algorithm gives lists which sweep out the object shape in a specific direction (clockwise), so the first step is to obtain a consistent starting point.

Considering the nature of the objects we are interested in modelling, i.e. pedestrians, we can make the assumptions both that there will be a consistent axis of maximum elongation and that these axes will be oriented in one direction. This is essentially just saying that people are very likely to be moving erect, standing on their feet. While this admits the possibility of fooling the model matching procedure by adopting a less usual orientation, it serves as a starting point.

The first step to reordering using this assumption uses the calculated value of the centroid point of the current shape. The centroid is simply the point whose cumulative distance from the known boundary points is a minimum. This has the physical analogue of the centre of mass of an object and was simply calculated during the boundary reordering by averaging the x and y coordinates of all boundary points.

Mathematically, for centroid c:

$$c = (\hat{x}, \hat{y}) \tag{4.30}$$

where:

$$\hat{x} = \sum_{i=0}^{n} \frac{x_i}{n} \tag{4.31}$$

$$\hat{y} = \sum_{i=0}^{n} \frac{y_i}{n} \tag{4.32}$$

and :

n is the number of points on the boundary

(x_i, y_i) are the coordinates of the ith point

This can then be used to find the principal axis of the object, which is the straight line passing through the centroid along which the sum of perpendicular distances to the boundary points is minimised. Using linear regression, the formula for the principal axis given by:

$$y = mx + c \tag{4.33}$$

slope m is given by:

$$m = \frac{s_{xy}}{s_{xx}} \tag{4.34}$$

where:

$$s_{xy} = \sum_{i=0}^{n} (x_i - \hat{x})(y_i - \hat{y}) \tag{4.35}$$

$$s_{xx} = \sum_{i=0}^{n} (x_i - \hat{x})^2 \tag{4.36}$$

Then the intersection with the y-axis, b is:

$$b = \hat{y} - m\hat{x} \tag{4.37}$$

This gives two points with which to specify the principal axis: $(\hat{x}, \hat{y})$ and $(0, b)$.

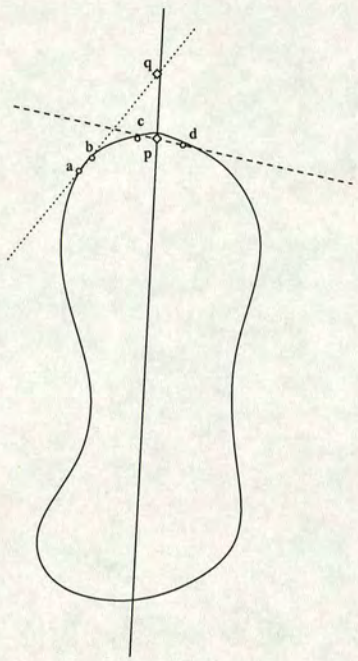

Figure 4.14: *Finding the intersection of the principal axis with a boundary. The line through points c and d gives the boundary intersection p between the defining points. The line through points a and b gives an off-boundary intersection q, away from the defining points.*

The procedure for this is fairly straightforward:

Initialse S_{xx}, Sxy at zero
    For all objects in boundary array
        For all points (x, y) in current boundary
        $S_{xy} = S_{xy} + ((x - \hat{x}) \times (y - \hat{y}))$
        $S_{xx} = S_{xx} + ((x - \hat{x}) \times (x - \hat{x}))$
    Intercept $= \hat{y} - (\frac{S_{xy}}{S_{xx}} \times \hat{x})$
    Store value in *intercepts_array*

To re-order the list, we select the point closest to the upper intersection with the boundary and the list is reordered with this as the first point. The intersection is found by considering each adjacent pair of points on the boundary in turn and the line that they define.

If the intersection between this line and the principal axis occurs on the segment of the line bounded by the two points, we have found a point of intersection with the curve which is a potential solution. For all other point pairs, the intersection will occur elswhere on the line (see

figure 4.14).

The solution with the minimum y coordinate constitutes the top most intersection and the boundary point to the immediate left of the intersection is used for the reordering.

Procedurally:

For all objects in boundary array
    Get the line defined by the current and next boundary points
    Find the intersection with the principal axis
    If the intersection lies between the current and next boundary points
        If the intersection has the lowest y value for any yet reported, store as current intercept
    Reorder the boundary with the new start at the point immediately anticlockwise of the stored intercept

Implementation points

A linear approximation is used to find the intersection point, which will not give a precise value for the true intersection. However, the value is only needed to sufficient precision to allow the choice of a consistent start point for every boundary. It is thus only neccesary to find the correct inter-point segment, as the start point is simply set as the anticlockwise bounding point. The approximation is unlikley to distort the final results.

As the top-most intersection is sought, it is possible to reduce processing by discarding points below the centroid (i.e. with larger y values) immediately.

4.6.2.3 Reparametrisation

The second step in generating a defining series of points with a consistent $1:1$ correlation between objects is to be re-represent the ordered list of points parametrically. The representation used is a piecewise polynomial which approximates the object's shape in a mathematically tractable form.

Restating the curves in terms of a single variable in this manner allows corresponding points to be selected and compared between boundaries of varied size and shape. For each such shape, all points can be specified by the value of this variable which can be set to span the same range around the curve irrespective of the boundary's absolute length.

We can then define the same number of equally spaced points (in terms of the base parameter) on each curve which hypothetically correspond to common points on a flexible object. The

mean position and variations from it can be calculated for a sequence of such shapes. Stating the curve parametrically also allows the mathematical formulation for a wider range of curves, including those which 'double back' upon themselves and so do not have a $(1 : 1)$ $(x : y)$ relationship.

Mathematically, in expressing a curve parametrically, we move away from the representation of the form $y = f(x)$ and look to specifying x and y coordinates separately in terms of some variable u i.e.

$$W(x, y) = W(u) = W(X(u), Y(u)) \tag{4.38}$$

Where $X(u)$ and $Y(u)$ are independent functions of u which specify corresponding x and y coordinates. The parameter u is chosen to increase along the curve from an initial minimum value at the start to a maximum at its end.

For our purposes we will be seeking the best fit to a given boundary list at all points and, as will be seen later, will desire that u steadily increases in value over the evaluation points. It is appropriate to define the parameter u in terms of distance along the curve from the start. Using the simplest measure of point-point separation, we consider the cumulative Euclidean distance between adjacent points on the curve. This takes the simplifying approximation of looking at straight lines connecting the points. The specification for u can be expressed as:

$$u_k = \begin{cases} 0 & : \quad k = 0 \\ \lambda \sum_{i=1}^{k} |W_i - W_{(i-1)}| & : \quad k > 0 \end{cases} \tag{4.39}$$

Where k numbers the points at which u will be evaluated and λ is a normalising function, calculated to set the length to N here, the desired number of control points for the curve approximation. W is defined for point (x_i, y_i) as:

$$W_i = \sqrt{x_i^2 + y_i^2} \tag{4.40}$$

In practical terms, this is implemented by calculating the Euclidean distance between each adjacent pair of points and, starting at zero for the first point, summing these to give the u value to which the x and y values at each point correspond. First, for each object, the ordered list of

boundary points is smoothed by applying 1-D Gaussian filtering:

$$G(x) = \frac{1}{2\pi\sigma} e^{-\frac{x^2}{2\sigma^2}} \tag{4.41}$$

This is implemented by applying a 1D kernel of the form:

K_1	K_2	K_3	K_4

where $K_1 = K_4 = 1$ and $K_2 = K_3 = 3$.

The Gaussian filtering gives a more even spread to the points at which the defining parameter is to be estimated. This is desirable to minimise any error caused by the interpolation process at any point along the boundary (error being proportional to the distance over which a curve is approximated by a straight line).

The reparametrisation involves one full loop over the ordered boundary list. At each point the cumulative Euclidean distance is calculated and stored in a 1-D array of the same size as the boundary array. After this procedure the array has distances stored in a 1-1 relationship with the boundary, ready for use in control point specification.

4.6.2.4 Control point specification

To approximate the curve, we will fit a sequence of cubic polynomials which approximate the local outline shape, specifically at 'control points' on the original boundary. The best fit criterion is specified in terms of minimising an error function of the form:

$$Error = \sum_{j=0}^{n-1} (P_x(u_j) - x_j)^2 + (P_y(u_j) - y_j)^2 \tag{4.42}$$

where the approximating curve is $\mathbf{P}(u) = (P_x(u), P_y(u))$ defined with respect to n control points. The jth control point has parameter value u_j and the control points on the original boundary against which the error is calculated are (x_j, y_j).

A simple choice could be made by choosing every N/nth member of the boundary list where N is the total number of coordinates in the boundary list and n is the desired number of control points, which for our closed curve will be equal to the desired number of spline sample points.

However, we wish the final control points to be at equal spacings in terms of u, in order to reduce computation requirements at a later stage (4.6.2.5). To achieve this we obtain total Euclidean distance from the final member in the distances array to calculate:

$$interval = \sum_{j=0}^{N} \frac{u}{n} \tag{4.43}$$

which is the increment in terms of parameter value we look for between adjacent control points. This is used to give the parameter values which correspond to the desired control points, by simply taking the integer multiples of $interval$ up to n. To find the corresponding $X(u)$ values we loop over the boundary list once more and again apply a linear approximation to the variation of $X(u)$ with u. So, taking that u values vary linearly between each successive pair of boundary points, we can employ linear interpolation to obtain:

$$X(u) = \frac{u - u_k}{u_{k+1} - u_k} X_{k+1} + \frac{u_{k+1} - u}{u_{k+1} - u_k} X_k \tag{4.44}$$

Using this formulation we can obtain a new set of $X(u)$ values at integer values of u from 0 to n. With a similar approach to $Y(u)$ we can form the desired set of points with fixed, approximately evenly spaced parameter values.

4.6.2.5 B-spline fitting

The curve which we are to generate, which will approximate the shape of the original outline at the control points specified above, is a *uniform cubic B-spline*. The overall curve is defined in terms of a sequence of polynomial pieces termed *segments* joined end-to-end.

The points at which the segments of a B-spline join are termed *knot points*, the parameter values thereat being referred to as *knot values*. Our knot points are the control points defined in section 4.6.2.4. By the reparametrisation process employed, the knot values K increase regularly and by our definitions we have set K :

$$K = \{k_0, \cdots, k_{n+n_{ord}}\} = \{0, ..., n + n_{ord}\} \tag{4.45}$$

where n is the number of control points and n_{ord} is the order of the B-spline $B_i^{n_{ord}}$ (here 3). In the case of a closed curve, the n_{ord} additional control points required to define the curve at say, the far end are repeats of the first n_{ord} points.

In considering how the standard B-spline basis functions combine with the given control points to give an approximating curve, there are two useful ways of considering the process:

- a linear combination (specified by the control points V_i) of cubic polynomial basis functions B_i
- a cubic-weighted (specified by the basis functions B_i) sum of control points V_i

To illustrate using the simpler case of linear approximation to a curve, we have already used the fact that any point between two vertices $X_i(u)$ and $X_{i+1}(u)$ can be expressed using linear interpretation as a weighted sum of the values $X_i(u)$ and $X_{i+1}(u)$. Equation 4.44 can be re-written as

$$X(u) = X_{i+1}B_{i+1}(u) + X_iB_i(u) \tag{4.46}$$

where $B_i(u)$ is a basis function, zero outside the range $u_{i-1} < u < u_{i+1}$ and which, graphically, constitutes a symmetric triangular *'hat function'* translated to and scaled over that range and centred on u_i.

Where, as in our case, the control points are equidistant, the basis function need only be scaled once and then simply translated to the appropriate interval to indicate the degree of mixing of two points at any point between them. This means that we can use a single B-spline basis function at any point to give the appropriate weightings of neighbouring points which generate the spline points. To give a smooth overall curve requires the use of cubic polynomial basis functions, having as they do positional and both first and second derivative continuity at the knots (termed C^2 continuity). To achieve this, the calculation of values for each segment requires mixing of two neighbouring points on each side, i.e. the basis function is non-zero over four successive segments (figure 4.15). This equally implies that four basis functions will be non-zero for any given segment (figure 4.16).

The derivation of the specific form of the standard B-spline basis function is given in [106] and gives a basis function over a particular segment as follows:

$$b_0(u) = -\frac{u^3}{6} + \frac{u^2}{2} - \frac{u}{2} + \frac{1}{6} \tag{4.47}$$

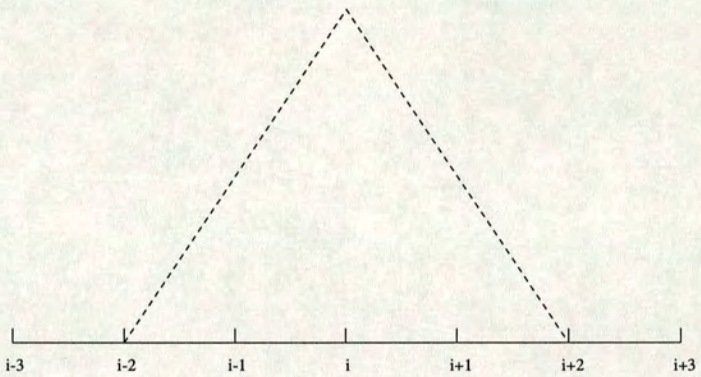

Figure 4.15: *A simple 'hat' basis function straddling four segments*

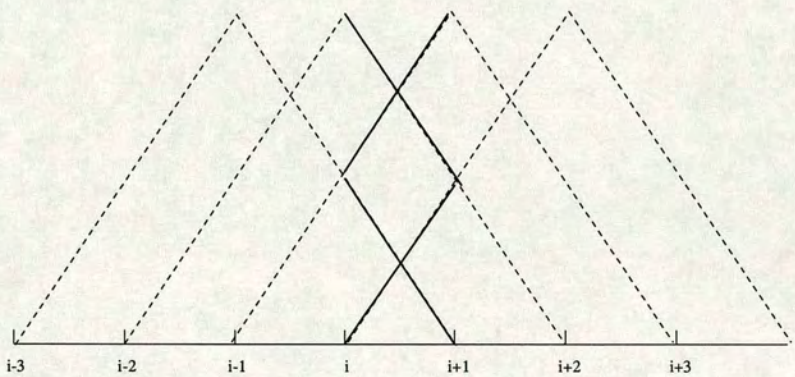

Figure 4.16: *The four basis function which are non-zero for a segment. The part of each basis function which intersects is shown solid*

$$b_1(u) = \frac{u^3}{2} - u^2 + \frac{2}{3} \tag{4.48}$$

$$b_2(u) = -\frac{u^3}{2} + \frac{u^2}{2} + \frac{u}{2} + \frac{1}{6} \tag{4.49}$$

$$b_3(u) = \frac{u^3}{6} \tag{4.50}$$

So, if we have four values $[x_0..x_3]$, the uniform cubic B-spline segment $[x_1, x_2]$, (where we set u=[0, 1] over that interval) will be:

$$X(u) = x_0.b_0(u) + x_1.b_1(u) + x_2.b_2(u) + x_3.b_3(u) \tag{4.51}$$

The expression for the overall spline curve $P(u)$ is:

$$P(u) = \sum_{k=0}^{N-1} V_k B_k(u) \tag{4.52}$$

i.e. the spline points are given by calculating for each control point, the basis function-weighted sum of the neighbouring control points as specified in the previous equations.

It is here that the importance of having control points with uniformly increasing parameter values becomes evident: we need only explicitly calculate a basis function for one segment of our spline and can re-use this for each successive segment to weight its neighbouring control points. Also, as we have established, the basis function for any point is non-zero over only four basis functions and so equation 4.52 reduces to

$$P(u) = \sum_{r=-3}^{r=0} V_{i+r} B_{i+r}(u) \tag{4.53}$$

Note finally that as we are approximating a closed curve, to enable the support for 'end' segments to cross the $u = 0$ point, we specify

$$B_k(u) = \begin{cases} B(u-k) & : \quad (u-k) \geq 0 \\ B(u+N-k) & : \quad (u-k) < 0 \end{cases} \tag{4.54}$$

So, the control sequence used is $(V_0, V_1, V_2 ..... V_N - 1, V_N, V_0, V_1, V_2)$ and the extra 3 control

points required are 're-used'.

The B-spline basis function calculation is implementable as a recursive procedure using the relationship of an $n_{ord}th$ order spline to an $(n_{ord}-1)th$ order spline and the base case of a single order (linear) spline.

The recursive relationship for $B_i^{n_{ord}}$ is:

$$B_i^{n_{ord}}(u_j) = \frac{u_j - k_i}{k_{i+n_{ord}-1} - k_i} B_i^{n_{ord}-1}(u_j) + \frac{k_{i+n_{ord}} - u_j}{k_{i+n_{ord}} - k_{i+1}} B_{i+1}^{n_{ord}-1}(u_j) \tag{4.55}$$

Where $i = 0, ..., n_{sample} - 1$.

The base case for first order spline B_i^1:

$$B_i^1(u_j) = \begin{cases} 0 & : \quad u_j < k_i \\ 1 & : \quad k_i \leq u_j < k_{i+1} \\ 0 & : \quad k_{i+1} \leq u_j \end{cases} \tag{4.56}$$

The recursive procedure is contained in the routine *blend* in the *VS_spline* program and is a standard implementation.

For our parametrically regular spaced curve, the basis functions at each point will be identical, which simplifies both the calculations necessary and the control structure of the *VS_spline* program. The reparametrisation and the specification of control points are both carried out during the construction of a *VS_spline* object:

For all objects in boundary array
- Build *distance_array* equal in size to current boundary
- For each point on the current boundary
  - Execute Gaussian filtering
- For each point on the current boundary
  - Calculate the cumulative Gaussian distance (g_i)
  - Store current (g_i) at corresponding position in *distance_array*
- Calculate required increment between points
- Use linear interpolation with *distance_array* and current boundary to generate n evenly spaced points

The calculation of the spline points themselves is a separate operation called for each required shape:

Initialise *parameter_accumulator* to zero

Calculate *increment* value required for n spline points at $\frac{(total-parameter-range)}{n}$
For all required spline points
    At all control points
        Use recursive blend function with current *parameter_accumulator* value to calculate current point position
        Add point to spline array
    increment *parameter_accumulator* by *increment*

Implementation points The spline fitting routine was adapted from an existing public domain routine *bspline.cpp* [2] intended for use in interactively fitting splines to screen images. The recursive bland function was lifted directly from this program and integrated with the in-house developed routines for reparametrisation and control point specification.

4.6.2.6 Eigenshape building by principal components analysis

The key to the usefulness of the flexible model approach is that it captures the oscillatory behaviour observed in flexible shapes such as pedestrians. The approach used to encapsulate this knowledge is to apply *principal components analysis (PCA)* to the covariance matrix of the representative sequence of uniform B-splines as constructed to this stage.

The central goal in PCA is the analysis of a distribution in multiple dimensions to derive an efficient re-representation of the relationships with the variables upon which the values depend. The process can be visualised by considering the points of the distribution to to lie in a dimensional space defined by n orthogonal axes, each of which corresponds to one of the base variables. In PCA, that n-1 dimensional hyperplane is plotted which captures the maximum variance in the distribution: in two dimensions this would be equivalent to plotting a line of best fit on a scatter diagram.

In multi-dimensional PCA, the process is repeated in sequential steps, plotting successive mutually orthogonal hyperplanes to capture progressively finer variations of the distribution. Taken to the logical conclusion, n hyperplanes may be plotted to capture all variation in the distribution: the important change is that they are ordered in terms of their significance in explaining the variation of the distribution.

Mathematically, the process is implemented by calculating the eigenvalues and eigenvectors of the covariance matrix of the sample distributions (here, spline point arrays). Here, the first eigenvector captures the maximum variance (indicated by its eigenvalue), the second captures

the next most variance and so on down to the nth. Relatively compact models can thus be constructed which explain most of the variation in the system but only using a subset of the eigenvectors.

To generate a model capturing the variance of a flexible human shape, the required input data is the sequence of B-spline approximations to the shapes of the training sequence. The mean shape and covariance matrix for the sequence is calculated and a standard module is used to calculate the eigenvectors and eigenvalues.

Allocate memory for *mean*, *posvar* and *temp* arrays (size $2 \times n$) and a $2n \times 2n$ *covar* array
For each spline in the training sequence
    For each point on the spline
        Add the point coordinates to the corresponding mean shape point
Divide each mean shape point value by the number of objects in the training sequence.
For each point on the spline
    For each spline in the training sequence
        Calculate difference from the corresponding mean point and store in *temp* array
    Accumulate values in *covar* array: covar[i][j] $+= \frac{temp[i] \times temp[j]}{numberofsplines}$
    posvar[i] = covar[i][i]

Note that the positional covariance estimate for each point is stored in the *posvar* array to augment the model. This is used in the model matching process.

Implementation points

Rather than apply effort to developing an in-house routine for calculating the eigenvalues and eigenvectors of the covariance matrix, an 'off-the-shelf' implementation of the *Jacobi Method* was used. This was used with no changes to the internal procedures, the only modifications being the minimum required to convert to C++ format and at the input/output level to give an efficient interface with the rest of the system. The implementation used was obtained from a standard source [1] and calculates the full set of eigenvalues and eigenvectors as two matrices (the first 1D, the second 2D) from input in the form of a **symmetrical** matrix. As the covariance matrix will always be symmetrical, this restriction is not a problem. The mathematical basis for the process and specifics of the implementation will not be detailed here, but are available in [1].

4.6.2.7 Model matching

The model matching procedure is itself a process with several distinct elements:

- Object segmentation
- Model Scaling
- Model Translation
- Calculation of normals to boundary
- Feature detection along the normals
- Minimisation of difference between predicted and observed points
- Evaluation of match score(s)

With the exception of the minimisation step, these are relatively straightforward routines. For the exception case, a novel method was implemented based around the *Simplex Method* for solving linear programming problems.

Object Segmentation

In the run-time detection process, the early processing mirrors that used in the model building procedure. The input to the model matching routine, apart from the model itself, is a labelled object map as detailed in section 4.5 which includes data on object centroid positions and object sizes. It is worth reiterating that the labelling module used in the matching process **must** be the same one used in the model building if scale measures are to be compatible.

Model Scaling

In attempting to match to a new object, the model is initialised as the mean shape component. This is scaled by taking the ratio of the new object size to the stored model size and applying this to each point coordinate of the mean shape.

Model Translation

The scaled mean shape is positioned by a simple addition to each point coordinate. The combined scaling and translation process can be expressed by:

$$x_i\prime = (S \times \hat{x}_i) + C_O + (S \times O_{\hat{x}}) \tag{4.57}$$

where:

$x_i\prime$ is the scaled, translated model point

S is the scaling ratio

$\hat{x}$ is the corresponding model mean shape point

C_O is the centroid position of the current object

$O_{\hat{x}}$ is the offset of the mean shape start point from its centroid

Normal Calculation

To asses the difference between the model predication and the actual object for each point on the model spline's curve, corresponding points on the object boundary must be specified. The chosen solution is to construct a normal to the model curve at each control point and to search for features in the object map along that normal.

The unit normal was calculated by first deriving the line connecting the two points immediately on either side of the current control point along the boundary. The normal is approximated as the line perpendicular to this which passes through the current control point.

A search window must be specified indicating how far along the normal features will be sought. This window size must vary from point to point in proportion to the range over which features can be anticipated in the normal flexing of the object. Taking the starting 'head' point to be fixed, this window size should vary along the boundary dependent on the characteristic vibrations in a pedestrian shape.

It is for this purpose that the pedestrian model was augmented by inclusion of a value for the average positional variance (σ_p^2) observed at each point, not a feature of the models used in [25]. If these variances are scaled up in proportion with the object/mean ratio (as in the Model scaling), they can be used as a basis to specify an appropriate search window size in terms of the standard deviation at each control point. A search window size of $2.5\sigma_p$ was found to be effective for locating sufficient points.

Feature detection

As the source image is a labelled object map, a trivial feature detection algorithm is acceptable. The feature sought is the outside edge of an object, irrespective of whether the object is specified by a blob or an outline. This is easily modelled as a step function from the object label value to background value. This is only slightly complicated by the need to allow for the predicted

point being within or outside the boundary. Features must be sought in both directions along the normal to allow for this, with the step function's profile being inverted dependent on direction. A simple weighting was applied so that the feature detected in either direction closest to the predicted point would be recorded.

A minimum threshold is applied at this point as the first stage in the matching decision: if features are not detected at more than half of the control points, the matching process can be failed here.

Simultaneous minimisation

The problem to be solved is the derivation of the combination of model eigenvectors which, when added to the existing scaled mean, will minimise the prediction error i.e. the overall separation between predicted and observed points. This is a complex simultaneous minimisation of connected variables: varying the amount of a particular eigenvector to be included will potentially affect the positioning of every point on the outline.

In the original implementation of this flexible model approach [25]an iterated *Kalman Filter* was used at each control point to track these differences using a B-spline interpolation matrix stored as part of the object model.

As noted, the chosen alternative approach here is a novel application of the *Simplex Method* used in solving linear programming problems. This method had been applied in previous unpublished work [107] in assessing similarity between iconic image objects. This suggested its applicability in this case as an interesting alternative approach to investigate.

In brief, the *Simplex Method* is a systematic approach to linear programming , which concerns maximising/minimising an *objective function* z :

$$z = a_{01}x_1 + a_{02}x_2 + ... + a_{0N}x_N \tag{4.58}$$

where

$x_p \geq 0$ for N independent variables : $p = [1..N]$

a_{0p} are scaling coefficients

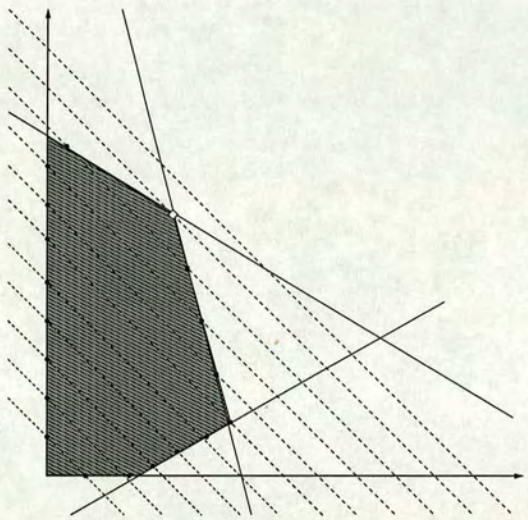

Figure 4.17: *Diagrammatic illustration of the Simplex method in two dimensions*

The objective function is subject to $m(= m_1 + m_2 + m_3)$ additional constraints, m_1 of the form:

$$a_{i1}x_1 + a_{i2}x_2 + ... + a_{iN}x_N \leq b_i \tag{4.59}$$

m_2 of the form:

$$a_{j1}x_1 + a_{j2}x_2 + ... + a_{jN}x_N \geq b_j \tag{4.60}$$

m_3 of the form:

$$a_{k1}x_1 + a_{k2}x_2 + ... + a_{kN}x_N = b_k \tag{4.61}$$

where:
$(b_i, b_j, b_k) \geq 0$
$i = [1..m_1]$
$j = [(m_1 + 1)..(m_1 + m_2)]$
$k = [(m_1 + m_2 + 1)..(m_1 + m_2 + m_3)]$
a_{ij} are scaling coefficients which may be positive, negative or zero

The concepts underlying the *Simplex Method* for solving such linear programming problems are well illustrated diagrammatically, as in figure 4.17. There, the solid lines correspond to

the constraints: the axes correspond to the $x_p \geq 0$ requirement, the other lines are m_1 type inequalities. The shaded region corresponds to the set of all allowed values which fulfil these constraints and the dashed lines correspond to selected solutions to the *objective function*, here simply $z = x_1 + x_2$. There are an infinite number of such solution lines possible and those chosen illustrate arbitrarily chosen integer solutions.

It can be shown [1] that the optimal solution can be obtained by considering the set of lines corresponding to the *objective function* and moving through them away from the origin by running along the boundary (the *simplex*) until the furthest vertex is reached.

On the diagram, the black circles along the boundary correspond to some possible solutions which are found during this process, the single white diamond at the vertex corresponds to the optimal feasible solution.

Considering the problem in these terms reduces the optimisation problem to combinatorial complexity for the n dimensional case. The *Simplex Method* constitutes a systematic procedure for investigating the combinations in n dimensions such that the objective function increases with each step and the number of iterations required can be guaranteed to be less than or equal to the larger of m or n

Reimplementation of the simplex method was not a productive use of resources and an 'off-the-shelf' implementation from [1] was again employed. The mathematical basis for the process and specifics of the implementation, not detailed here, are available in [1].

The main challenge in adopting this solution was to restate the simultaneous minimisation problem in a linear programming format. The central aim is to minimise the overall distance between the set of n_p pairs of predicted and observed points, d where:

$$d = \sum_{i=0}^{n_p} d_i \tag{4.62}$$

and d_i is the distance between the ith pair of points:

$$d_i = f_i - p_i \tag{4.63}$$

where:

f_i is the ith observed point coordinate

p_i is the ith predicted point coordinate

The set of predicted points can be varied by altering the amount of each eigenvector which is included in the current prediction:

$$\mathbf{p} = \hat{\mathbf{x}} + \mathbf{b} \times \mathbf{E} \tag{4.64}$$

where:

$\mathbf{p}$ is a $(1 \times m)$ vector giving the current prediction

$\hat{\mathbf{x}}$ is a $(1 \times m)$ vector giving the scaled model mean

$\mathbf{b}$ is a $(1 \times m)$ vector specifying the amount of each eigenvector to be included in the current predication

$\mathbf{E}$ is a $(m \times n)$ array containing the m eigenvectors, each of size n

Considering the ith predicted point, p_i:

$$p_i = \hat{x}_i + \sum_{j=0}^{m} b_j \times e_{ij} \tag{4.65}$$

where j indexes the m eigenvectors. Combining equations 4.63 and 4.65, we get:

$$d_i = f_i - \hat{x}_i - \sum_{j=0}^{m} b_j \times e_{ij} \tag{4.66}$$

and substituting into equation 4.62, we get:

$$d = \sum_{i=0}^{n_p} (f_i - \hat{x}_i) - \sum_{i=0}^{n_p} \sum_{j=0}^{m} b_j \times e_{ij} \tag{4.67}$$

which we can rewrite summarily as:

$$d = d_{static} - d_{variable} \tag{4.68}$$

where d_{static} is a constant term expressing the difference between the observed feature points and the scaled model mean and $d_{variable}$ is the variable component which we wish to optimise. The variable which we control is b_j, the amount of each eigenvector to be included in the current

predication, so we rewrite as:

$$d_{variable} = \sum_{j=0}^{m} \sum_{i=0}^{n_p} e_{ij} \times b_j \tag{4.69}$$

$$= \sum_{i=0}^{n_p} e_i \times b \tag{4.70}$$

So we now have:

$$(d - d_{static}) = -\sum_{i=0}^{n_p} e_i \times b \tag{4.71}$$

which is in a form amenable to application of the simplex method.

Vector array b is the list of variables $(x_1, ..x_N)$ and the coefficient a_{0i} of **each** is the sum of the elements of the corresponding eigenvector. This constitutes the *objective function* to be minimised.

The primary constraints are that the difference at each point must individually be below a specified margin threshold, M_t:

$$|d_i| < M_t \tag{4.72}$$

Here, the scaled positional variances can again be used as the basis for the threshold values ($2.5 \times \sigma$ again).

As the unmodified minimisation process would push toward the highest negative value of z, an overall secondary constraint to keep the minimised function non-negative had to be added:

$$d \geq 0 \tag{4.73}$$

At this point it became clear that there was no straightforward way to handle the case of the ideal optimum solution being a small negative value within the single Simplex run. A second 'mirror image' run is needed with primary constraints and the objective function remaining

unchanged but the task being a maximisation subject to a revised secondary constraint:

$$d \leq 0 \tag{4.74}$$

The result with the lowest magnitude from the two runs is accepted as optimal.

Match Result

The overall match decision is comprised of three component parts. Firstly, during the feature detection phase, more than half the control points must find corresponding features. If this is not satisfied, perhaps due to severe occlusion of the object, the match fails there. Secondly, the predicted/observed differences are used as input for the Simplex routine: if no solution is found subject to the imposed constraints the match again fails.

Finally, the optimised result must be above a specified threshold: the maximum value to which this can be set is simply the sum of the positional variances for the individual points, the minimum zero, demanding an exact match. This threshold is set empirically based on initial test results and the maximum value was found to be suitable.

Where a satisfactory overall result is obtained, by the structuring of the test, it is also then known that the result has a satisfactory overall difference and that no individual point is distorted by more than a second specified amount. The object can then be stored as a confirmed pedestrian and the requisite update procedures performed.

Implementation points

For use of the standard *Simplex Method* routine, input data must be arranged in a quite specific format, a *tableau* implemented as a 2D array $a[i][k]$. Row one contains the *objective function* coefficients and the balance of rows the coefficients of the constraint equations in the strict order m_1,m_2,m_3.

Further, the array will not be of a fixed size between match sessions as there may be a variation in the number of features located. A tableau building routine was needed which dynamically adapted size and overall threshold to reflect the number of points. A second such routine built the tableau for the 'mirror image' maximisation, which in this implementation is run as a minimisation problem with constraints manipulated to simulate the 'negative problem'.

The output of the routine is the transformed tableau array plus a 1D array indexing which row now corresponds to which variable. The indexing was used to give the desired list of *deltas*

indicating the amount of each eigenvector included in the optimum predication model.

Use of the *Simplex Method* to solve this problem relies on several simplifying assumptions concerning the system. A central assumption is that the variables are independent which, as they constitute points on a continuous line is not strictly the case. The predicted points are allowed to vary independently here, which diverges from a strict realism.

While the resulting shape could potentially be more irregular than is entirely feasible, the effects of this are mitigated as a side effect of the imposition of absolute maxima on the individual divergences.

The need to specify threshold values introduces assumptions as to what is to be considered acceptable for the system. The specification that more than half the control points must generate observed features will inevitably reduce the system's ability to handle severe occlusion.

Use of the scaled model's positional variances to give the allowed deviation at each point is intuitively appealing, but could lead to point loss where extreme motion occurs.

4.6.2.8 Pedestrian objects and arrays

When an object is identified as a pedestrian, a new *VS_pedestrian* object is constructed. Each has a Unique Identification Descriptor (UID) allocated which allows the pedestrian to be specified for tracking and retrieval purposes. The object age in number of frames is set at 1 and incremented for every frame where the pedestrian is matched to an object. The pedestrian data also includes an array which records the current and historic centroid positions and which thus specifies its historic trajectory which can be used for analysis purposes.

Two prediction arrays augment the pedestrian data, one giving an estimate of the pedestrian shape in the next frame using a simplified *Kalman filter* prediction of the shape parameters (see section 4.6.2.9) and the other containing an updated estimate of shape parameter variance.

When a pedestrian object is constructed, it is stored in an array of *VS_pedestrian* objects which contains all such entities found and not yet lost at the current frame. At the start of processing of each new frame's labelled object map (before located objects are matched against the scaled median shape) matching is attempted with the extant objects from this array.

The prediction arrays are used along with the tracking module to generate a scaled, translated

shape, pre-distorted from the scaled mean using the predicted distortion parameters. This is used to match against the object using the same criterion as are applied for new objects against the scaled mean shape.

The relative positions are evaluated first, in terms of the known centroid points and pedestrians outside a specified distance from the object are discounted. The pedestrian with predicted position closest to the object's actual position within the distance threshold and which matches the object's shape is taken as the positive result.

The choice of threshold distance does not depend upon evaluation of the maximum pedestrian velocity as this is modelled in the Kalman filter tracking (section 4.7): the distance is solely a measure of the allowed positional prediction error. In the current implementation a Euclidean distance of 100 pixels was fixed during testing. To cope better with different image scales, it would be better to vary this in proportion to the estimated scene:image ratio.

A Kalman filter is maintained to predict position of the pedestrian and its scale as described in more detail in section 4.7

4.6.2.9 Simplified Kalman filter predictions

The shape prediction array for each pedestrian object is made using a simplified Kalman filter for each shape parameter b_i, initialised at the value calculated for the first match against the scaled mean shape. The companion variance prediction array contains the estimate measurement variances for each parameter, initialised at the scaled positional variance used with the scaled mean shape. The *Kalman Gain*, g_i for each parameter is calculated using:

$$g_i = \frac{c_i}{c_i + v_i} \tag{4.75}$$

where:

c_i is the variance of the ith shape parameter, b_i, initialised at the ith eigenvalue

v_i is the measurement variance, estimated for the ith point using the average of the scaled mean shape positional variances

The parameter predictions are then calculated using a simplified filter equation:

$$b_{i1} = b_{i0} + g_i(f_i - b_{i0}) \tag{4.76}$$

where:

b_{i1} is the new parameter prediction

b_{i0} is the old parameter prediction

f_i is the corresponding observed value

The error covariance is updated using:

$$c_{i1} = (1 - g_i)c_{i0} \tag{4.77}$$

where:

c_{i1} is the new error covariance estimate

c_{i0} is the old error covariance estimate

4.6.3 A novel pixel-based object classification approach

The second module option was a novel *pixel-based approach* developed to complement the *model-based* technique. To provide the desired contrast the method should involve a minimum of mathematically complex elements. This was doubly appropriate as this alternative approach should also focus on minimising the resource requirements demanded of the object classification process.

As noted, *pixel-based* approaches rely on basic geometrical properties to identify objects, whereas it is clear that moving pedestrians are not in themselves simple geometrical shapes. It was therefore necessary to make assumptions and approximations in attempting to design a pedestrian detection module. The assumptions were made from first principles, based on *a priori knowledge* of the attributes of pedestrians and their relationship to surveillance scenes. The assumptions used in the approach were as follows:

1. Position can be represented by object centroid (this assumption is used in the *model-based* approach also).

2. Orientation can be represented by object principal axis. This assumes that a pedestrian will generally be erect while in motion.

3. Shape can be approximated by a rectangle or ellipse. This allows the ratio of lengths parallel (l_{par}) to and perpendicular to (l_{per}) the principal axis to define the object shape.

4. Pedestrian shapes, expressed by ($l_{par} : l_{per}$) will not usually fall below a certain threshold value.

Using these assumptions, a procedure for detecting pedestrians using a shape-based approach, from an input in the form of a labelled object map (section 4.5) could be developed:

1. Boundary extraction

2. Calculation of the principal axis

3. Shape estimation

4. Shape classification

5. Registering and listing shape objects

4.6.3.1 Boundary Extraction

The process for boundary extraction is held in common with the model -building approach and is as detailed in section 4.6.2.1.

4.6.3.2 Principal axis calculation

Calculation of the principal axis involves minimising the sum of perpendicular distances to the boundary points and is implemented using the linear regression procedures detailed in section 4.6.2.2.

4.6.3.3 Shape estimation

Shape estimation is performed by finding l_{par} and l_{per} to specify the object's principal axis oriented bounding box. This approach is equivalent to specifying the shape using its elongation

value, E, where:

$$E = \frac{\chi_{max}}{\chi_{perp}} \tag{4.78}$$

and:

χ_{max} is the length of the shape along its axis of orientation, the principal axis calculated in section 4.6.3.2

χ_{perp} is the length of the shape along the axis perpendicular to χ_{max}

The true elongation uses χ_{min}, the length of the shape along the axis giving the minimum results in place of χ_{perp}

A heuristic approach is taken to estimating E, as follows:

```
Begin at object centroid point
    For both of the specified axes
        In each direction
            Find the first point outside the object, set previous point as an initial bound point
For each bound point
    Move one point along the axis (away from the centroid)
        Check perpendicular to this point
            If any object point is seen within range of current bound box
                move out another point along the axis and repeat
            else, set this as the new bound point
```

The result is as illustrated in figure 4.18: the initial bound points lie on the object boundary and are pushed outward to the final points shown, giving a bounding box aligned along the object's principle axis.

4.6.3.4 Shape classification

A simple elongation thresholding is used to to classify objects. A shape is classed as a potential pedestrian if the elongation as calculated in equation 4.78 exceeds a specified value. The threshold ratio is set at 1.5 drawing on contextual knowledge, implicitly assuming that pedestrians correspond to long, narrow objects.

The thresholding can be extended to cover pose using a supplementary elongation calculation. Given that it is now known that the shape is within acceptable limits, a second bound box can be estimated using the four boundary points with the maximum or minimum x or y coordinate

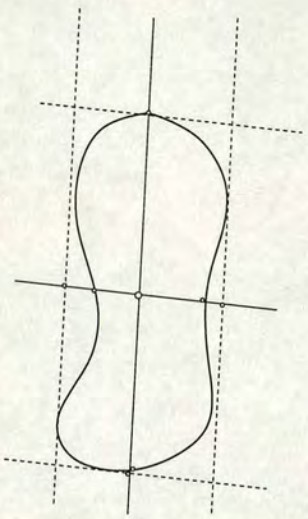

Figure 4.18: *Finding the bounding box. The solid straight lines are the two axes through the centroid point. Each small circle is an initial or final bound box located along an axis. The dashed line is the corresponding bounding box*

values respectively. This box is aligned with the vertical and if its elongation is above a second threshold, the pose will be correspondingly close to vertical.

Two examples are shown in figures 4.19 and 4.20 to illustrate the extreme cases of a near vertical object (a potential pedestrian as used in figure 4.18) and a near horizontal one. To allow for some variation in pose, the threshold was set at 1 for this bounding box.

4.6.3.5 Shape objects and arrays

When an object is identified as a pedestrian, a new *VS_shape* object is constructed. Very similar to the *VS_pedestrian* objects described in section 4.6.2.8, these also have UIDs, age, centroid array and a positional/scale Kalman filters. The distinction is that there is no need to maintain arrays for prediction here.

Again like the *VS_pedestrian* objects, these are stored in an array of all such entities found and not yet lost at the current frame. The same approach is used to identify extant *VS_shape* array members with newly discovered objects although the process here is simpler as no matching is required. If the object has been classified as a pedestrian, the shape with predicted position closest to the object's actual position within the distance threshold is taken as the positive result.

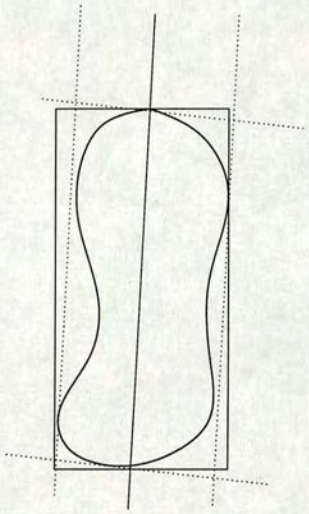

Figure 4.19: *The second bound box compared with the original object in the same pose*

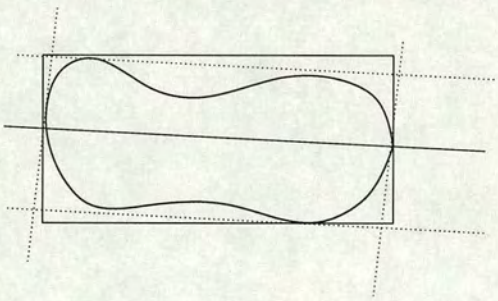

Figure 4.20: *The second bound box compared with the original object at an extreme variant pose*

4.6.4 Functionality testing

4.6.4.1 Model-based approach

The procedures for model fitting were assessed by visual review of image-like output of runs on short sequences of artificial test images, an excerpt from which is shown in figures 4.21 and 4.22. The various stages of the model matching process were tested incrementally, using labelled object maps as input, with output primarily in the form of image-like representation, supplemented by textual log output where required.

The boundary extraction was tested by image-like output of the boundaries projected onto the labelled object map.

The reordering process was examined by generating textual lists of the boundary points before and after the reordering process, along with details of size and centroid. The linear regression calculation procedure had been pre-tested on text book problems with known answers prior to integration.

B-spline fitting was assessed visually, by adding an image-like output of the spline points to the test image sequence, an excerpt from the results being shown in figures 4.23 and 4.24. Model translation and scaling, normal calculation and feature detection were all assessed similarly by superimposing the calculated results on the test image sequence. An excerpt showing illustrating the centroid fitting stage is given in figures 4.25 and 4.26.

As an off-the-shelf unit, the *Jacobi routine* function had been pre-tested by use in multiple environments and as such, the correct basic operational structure was treated as known. The correct functionality was double checked by testing on text book problems (for calculation of eigenvalues and eigenvectors) with known answers.

The implementation of the *Simplex method* was again an off-the-shelf unit and so the correct basic operational structure was treated as known. The correct functionality was double checked by testing on text book problems (for calculation of simultaneous optimisation solutions) with known answers.

Object identification was tested on short sequences of artificial test images. A model trained from a sequence of such shapes was used for matching. Identification was signalled in the visual output by marking the centroid position of each identified object. Textual output of the

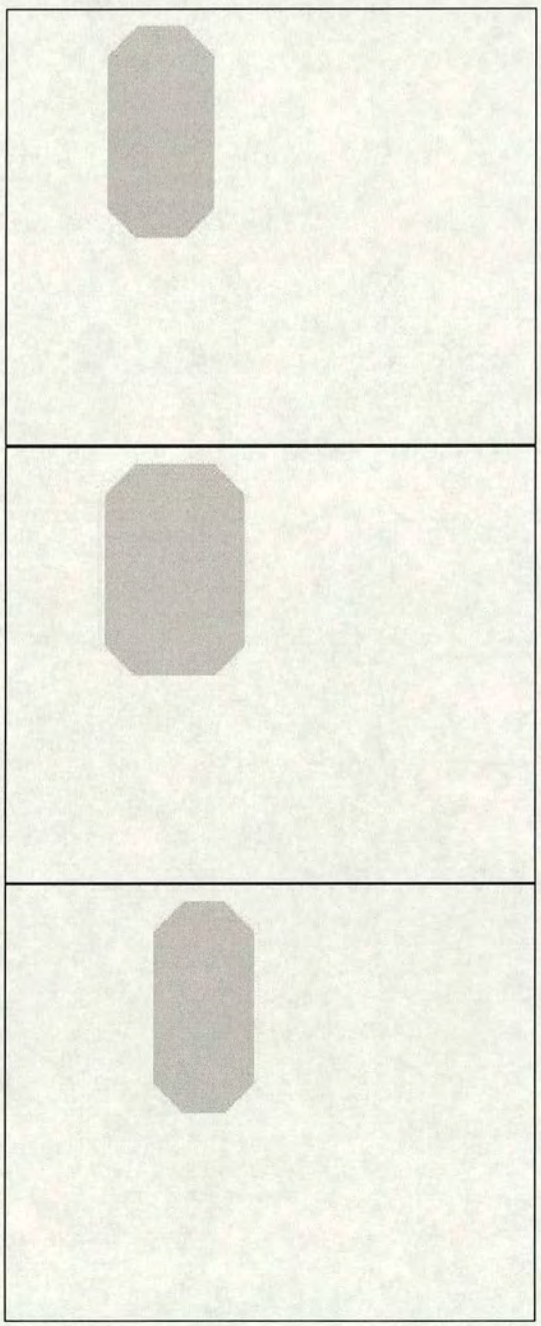

Figure 4.21: *Excerpts from the artificial test image sequence (i)*

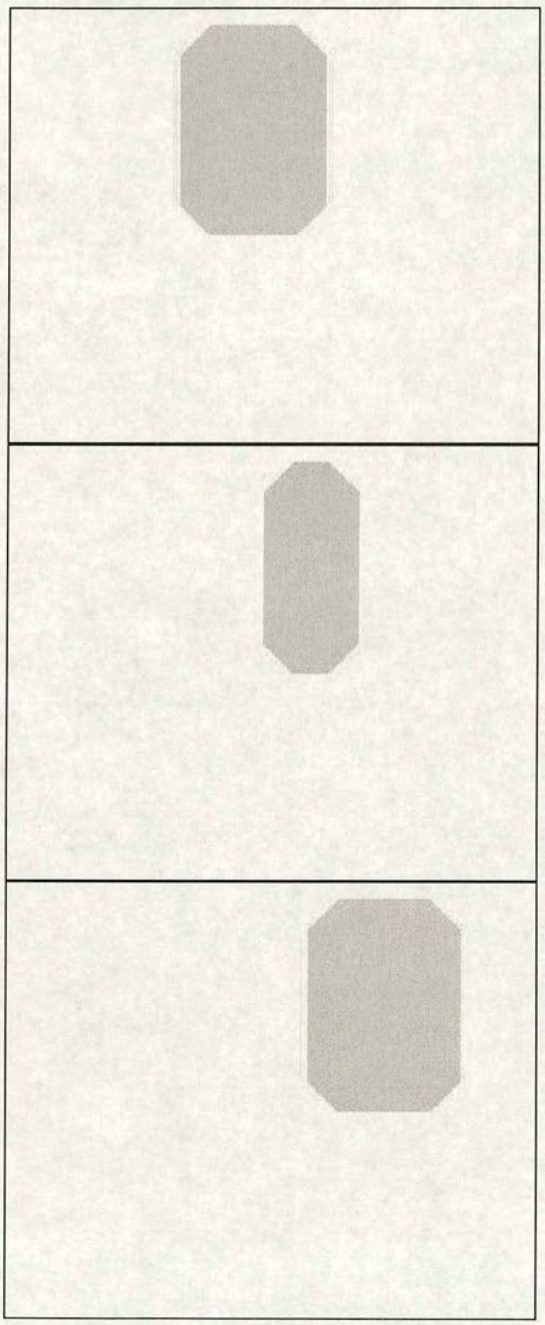

Figure 4.22: *Excerpts from the artificial test image sequence (ii)*

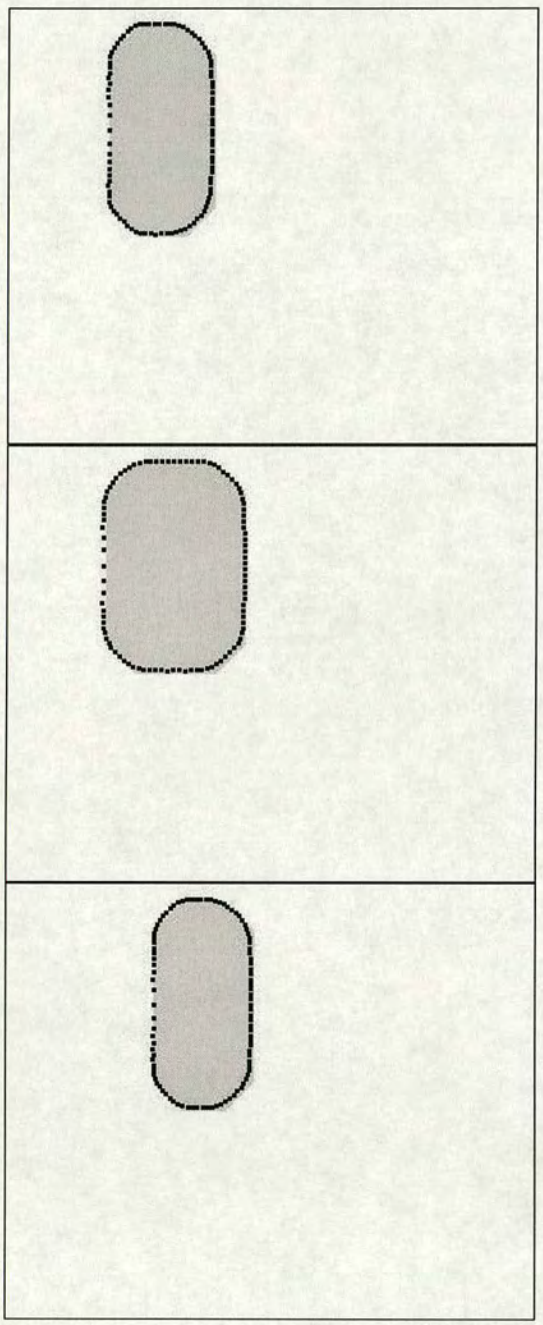

Figure 4.23: *B-splines fit to the artificial test image sequence (i)*

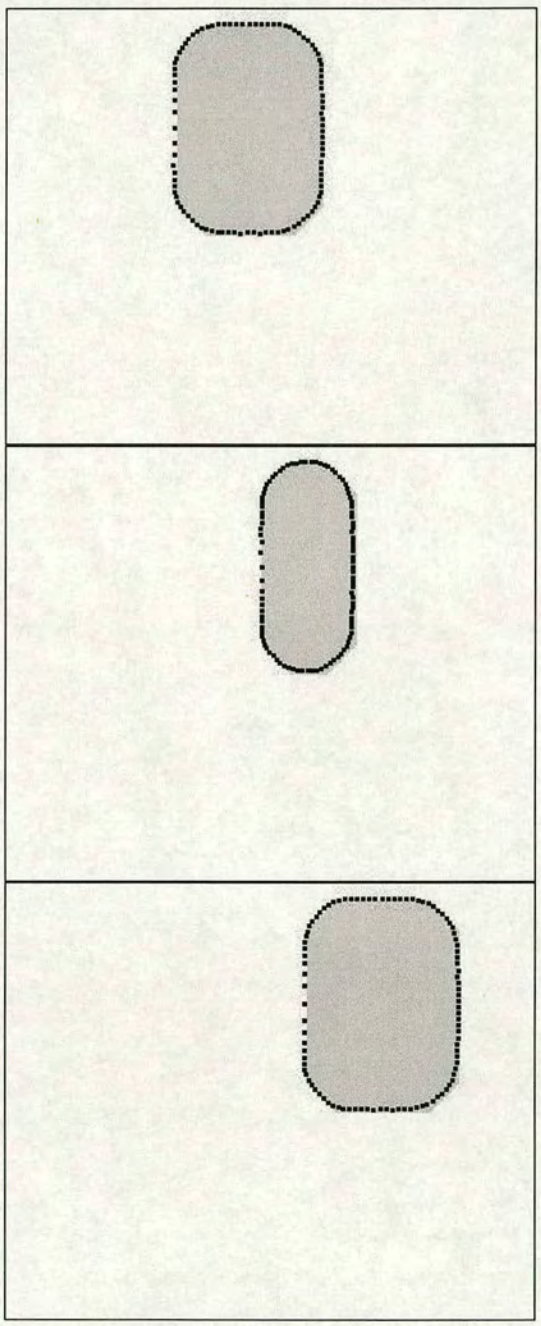

Figure 4.24: *B-splines fit to the artificial test image sequence (ii)*

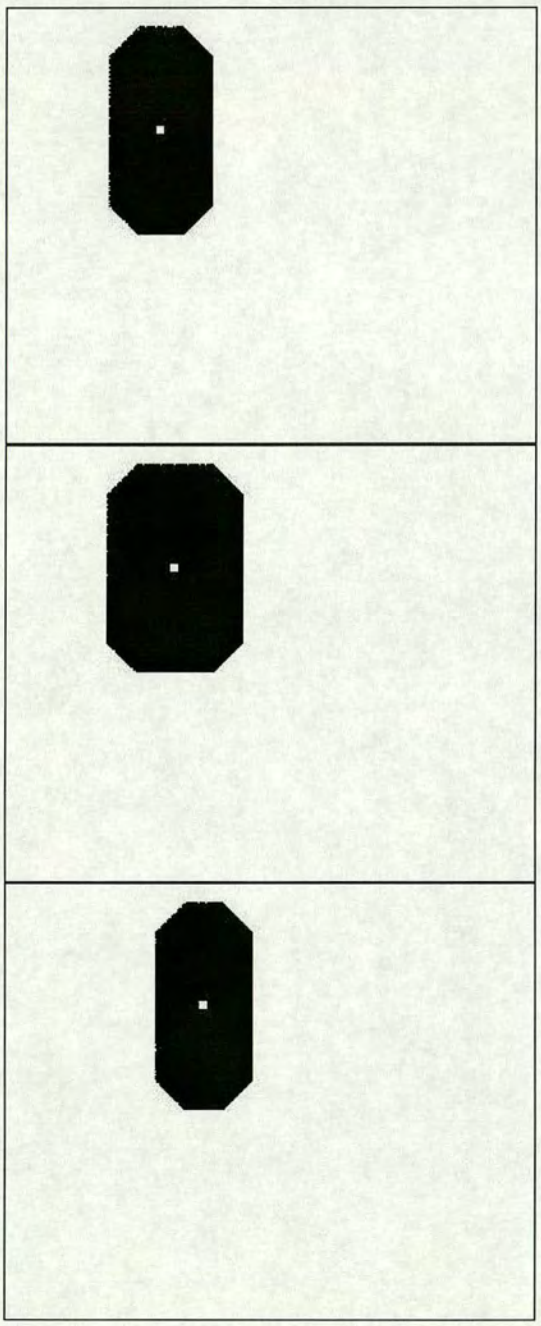

Figure 4.25: *Centroids showing model fit to the artificial test image sequence (i)*

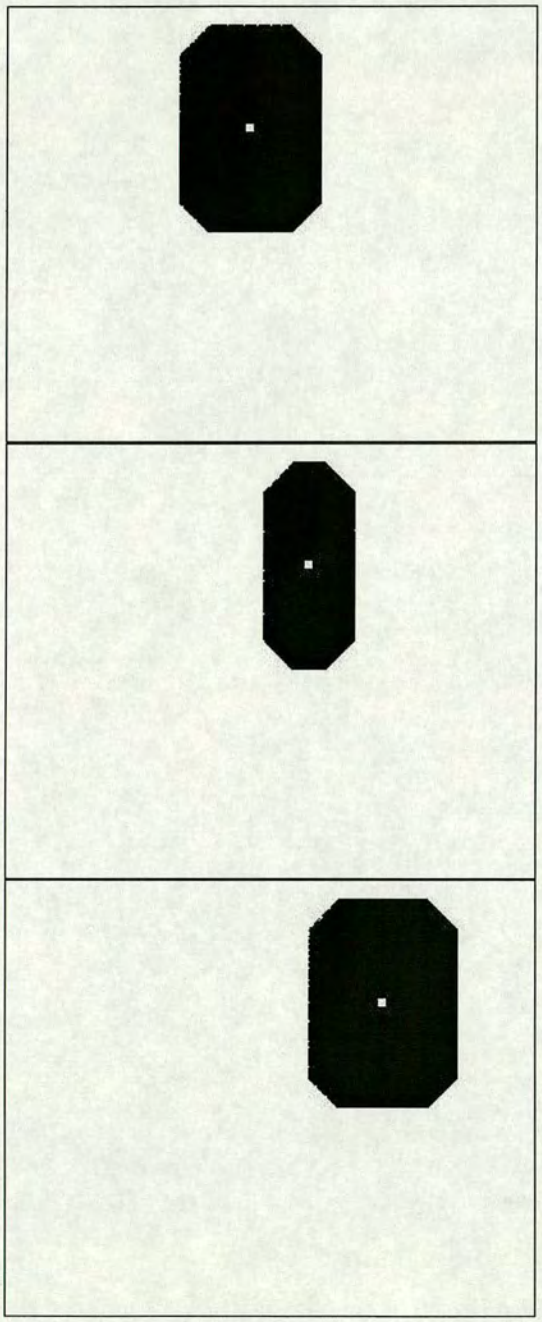

Figure 4.26: *Centroids showing model fit to the artificial test image sequence (ii)*

centroid history, UID and age of each matched object was provided as supplement to assist resolving potential ambiguities.

4.6.4.2 Shape-based approach

The boundary box calculation and subsequent classification were tested in the form of image-like representation, supplemented by textual log output where required.

Object identification was tested on the same short sequences of artificial test images and identification was signalled in the visual output by marking a cross joining the bounding points on each identified object, as illustrated in figures 4.27 and 4.28 . Textual output of the centroid history, UID and age of each matched object was provided as supplement to assist resolving potential ambiguities.

4.7 Tracking

As noted in section 4.1, only one solution to the tracking task was implemented in the current testing because tracking is essentially an extension after the first possible alarm stage and can be implemented from an extant module. The tracking module is included both to check on the usefulness of the output of model classification as input to a succeeding module and to facilitate the development of using trajectory data for analysis (section 4.8.2).

An 'off-the-shelf' (in-house) developed tracker was adapted to integrate it with the output of the subsystem to this point. This was both to assess the system's compatibility with such an independently designed unit and to avoid focusing excessive development time on this 'support' module. The implementation used was the *VS_Kalman* routine developed by Andrew Peacock for the University of Edinburgh Department of Electronics and Electrical Engineering's Vision Group library.

A full discussion of the theory underlying the Kalman filter will not be presented here as no modification or development work was carried out on the routine other than integration with the system. Further, the performance characterisation results (Chapter 6) are not extended to examine the tracker's performance by the same rationale.

In brief then, for a simple 1D version of the filter, where the filter's current state is equivalent

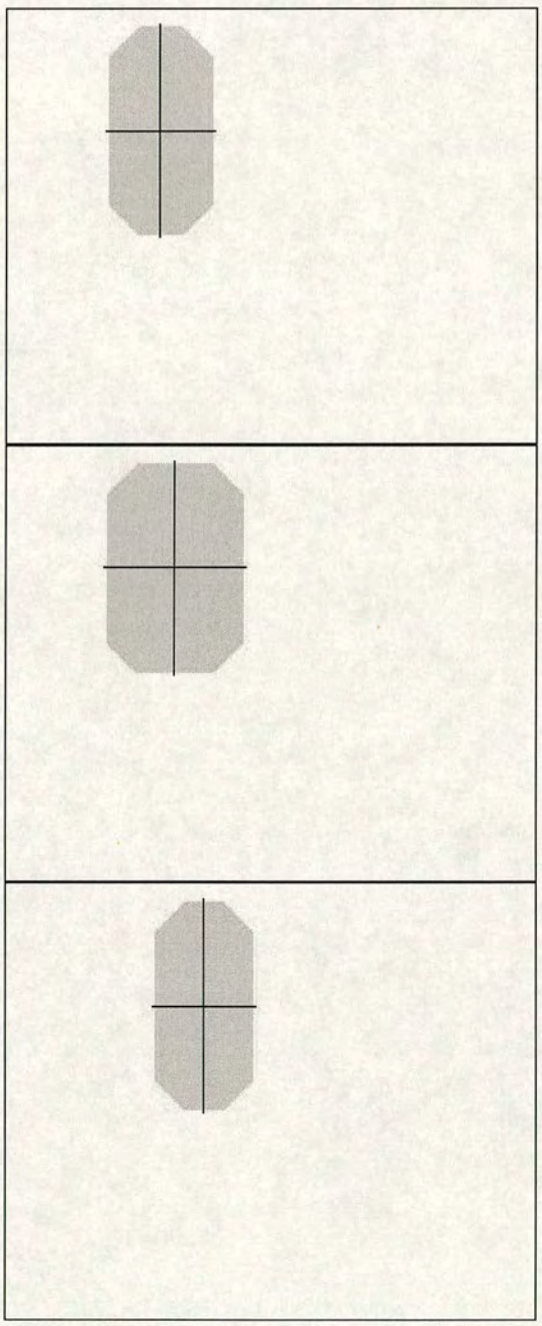

Figure 4.27: *Bound box axes fit to the artificial test image sequence ((i)*

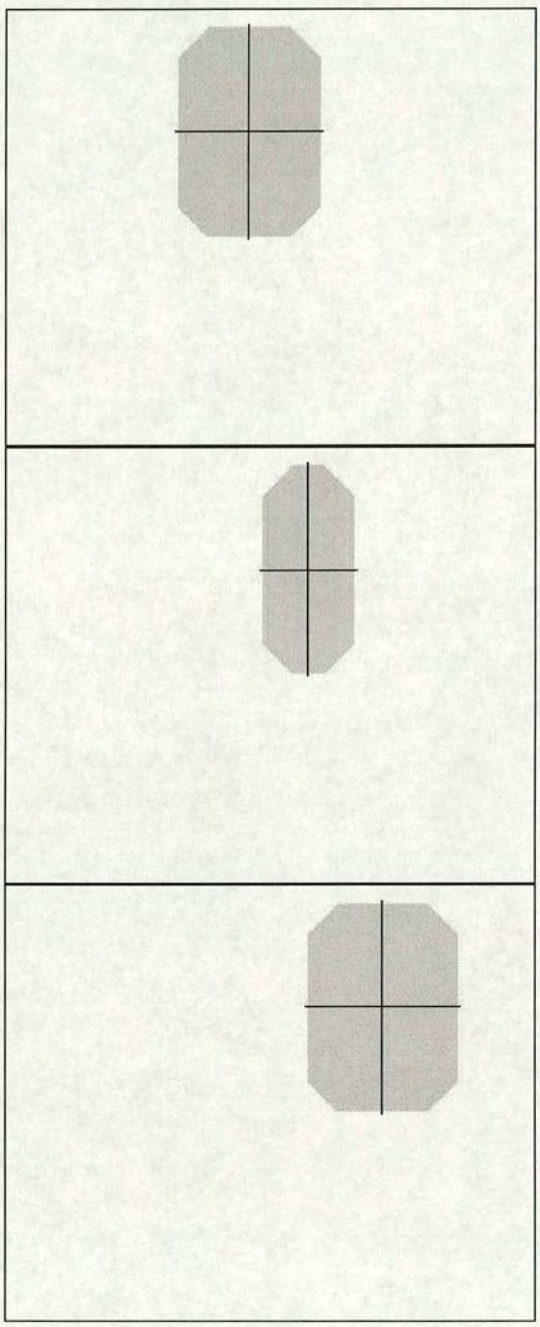

Figure 4.28: *Bound box axes fit to the artificial test image sequence (ii)*

to a direct estimate of the measured variable, the algorithm steps are as follows:

1. Input a new measurement (x_m) and estimate error covariance (c)

2. Calculate the Kalman gain, g_k using:

$$g_k = \frac{c}{c+v} \tag{4.79}$$

   where v is measurement noise variance

3. Re-estimate using:

$$x_1 = x_0 + g_k(x_m - x_0) \tag{4.80}$$

   where:

   x_1 is the new estimate

   x_0 is the old estimate

4. Re-estimate error covariance c using:

$$c_1 = (1 - g_k)c_0 \tag{4.81}$$

   where:

   c_1 is the new estimate

   c_0 is the old estimate

5. Make predications of the process state, x_p, c_p using:

$$x_p = A \times x_1 \tag{4.82}$$

$$c_p = A \times c_1 + Q \tag{4.83}$$

   where:

   A relates the process state at time $t + 1$ to that at time t

   Q is the variance of any process noise

A full Kalman filter allows for complex relationships between state, measurement and control input and can allow for multidimensional variables.

For the straightforward application to the tracking system, the velocity of centroid movement and rate of change of size were the states modelled by separate Kalman filters.

4.8 Extensions

As noted, the system is designed with the possibility of extension modules being integrated at a later date as a significant consideration. There are three specific modules under development for inclusion with the system to extend its abilities. The first of these, face capture is the most interesting as it is a module which can be activated for all pedestrians or as a result of an alarm only. In this respect, it could be run from the central control point and offer the additional functionality without a corresponding overhead in resource usage.

4.8.1 Face capture

For surveillance applications, identification of individuals in real time or from stored footage is a significant issue. The ability to capture face images in compact form for further analysis alongside extant alarm data would be valuable in the final application.

On the final platform, this could be implemented using a camera's control input to automatically direct pan, tilt and zoom (PTZ) to capture a high definition face image. Using a simpler camera, a sub-image corresponding to the head position could be extracted for review and/or longer term storage.

It is possible to implement the latter approach on the test platform. In both classification modules, the centroid, principal axis and bounding box of the pedestrian are ascertained. Using this data and assumptions as to the standard construction of a pedestrian drawn from contextual knowledge, sub image capture can be simply implemented.

A simple average pedestrian layout is specified with the ratio *(head:trunk:legs)* set at *(1:2:2)* (see diagram 4.29). Using this model, a sub-image of the top 20% of the bounding box is likely to capture the head region. Although not guaranteed or exact, the saving of storage is considerable: even storing just the whole bounding box requires 400% more space.

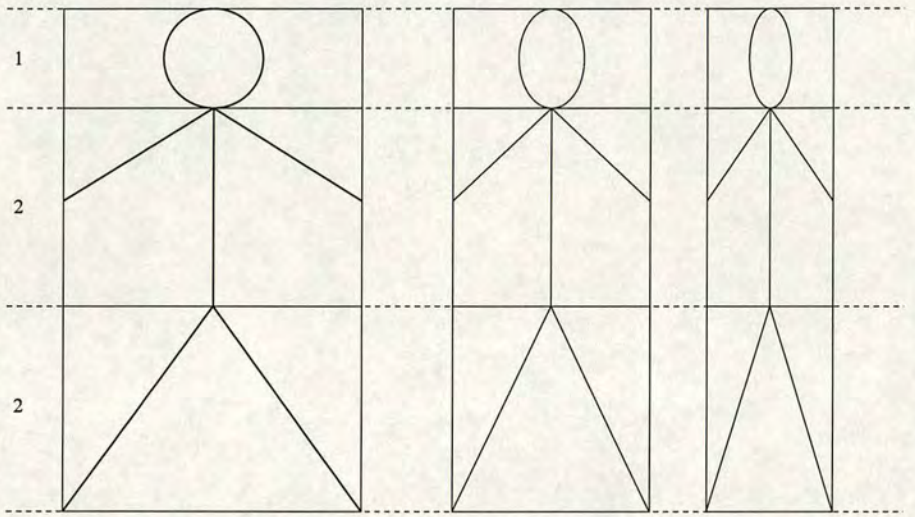

Figure 4.29: *An illustration of the 1:2:2 ratio assumption for three different pedestrian stances.*

As no face recognition software should be assumed, the sub-image captured will not be evaluated during capture for quality (i.e. the front or back of the head may be captured). It would be prudent to capture a sequence of these sub-images from the objects history to maximise the likelihood of a good face shot being obtained.

An illustration of the operation of a simulated version of this extension is given in section 6.5.

4.8.2 Trajectory extraction

The trajectory data for each identified pedestrian is recorded as part of *VS_pedestrian* or *VS_shape* objects. A simple extension is to store these trajectories either in isolation or alongside face sub-image sequences (see 4.8.1). In [32], such data is analysed to extract the distance to whichever is the nearest vehicle at a given time: at the point of closest approach to each vehicle, the distance and instantaneous speed were recorded and plotted as a scatter plot for all vehicles.

Taking the measurements over all trajectories observed they then built up a 2-D cumulative probability distribution. The cumulative probability of the speed-distance tuple at each closest point would be used as a measure of interest: the closer/slower to a vehicle, the more interesting the event.

A module to perform this kind of analysis could be implemented as a direct extension of the current system, but a simpler alternative is possible. If the trajectory data were stored alongside

the face sub-image sequences with both labelled using the UID of the corresponding object, the two taken together would provide evidence linking identity data with a summary of trajectory behaviour.

Furthermore, this alternative provides an avenue to solving the problem of analysing the behaviour of an individual over several visits to a scene. If the behaviour is stored with sufficient identifying data (the face sequences) then atypical behaviour analysis can be made for the group of scene visits overall, taking into account number of returns in a specified time-span, trajectories executed in each etc.

This type of behaviour analysis is of particular interest in analysing the behaviour of shoplifters, car thieves and other criminals performing reconnaissance and making frequent returns in an attempt to conduct their business unobserved.

4.8.3 Behaviour analysis

Both the existing plan for alarming on entering a restricted area and the proposed face/trajectory review constitute basic behaviour analysis procedures.

The system provides data which is suitable for input to a wide range of more complex behaviour analysis methods which would need to be evaluated within the context of a larger composite system. Possibilities include the use of Bayesian analysis to generate semantic descriptions of behaviour [86] or to interpret behaviour patterns [108], building spatiotemporal models to predict/evaluate human/human and car/human interactions [105] and using information fusion to combine the system results with other data sources [109].

4.9 Summary

This chapter has presented the motivations for choosing the specific module options to be investigated. The theory underlying significant modules and submodules has been presented where it has relevance to performance evaluation and design.

The implementation details have been presented at a level appropriate to indicate control choices and allow re-implementation without presenting code-level specifics. The evaluation procedures used to ascertain correct functioning and assist in setting tuning parameters have been

summarised for each module.

The overall control structure has been described and proposed extensions which are appropriate within that control structure (and which require relatively little additional work/on-line resources) have been introduced.

Now that the design and implementation of the system have been specified, it is necessary to find a suitable performance characterisation approach and to obtain results sufficient for selecting the optimum system variant to be implemented on the final platform.

Chapter 5
Performance Characterisation

5.1 Introduction

The performance characterisation of the alternate approaches for the final application is a vital step in constructing a system which is appropriate and performs optimally both in terms of processing abilities and resource requirements. There is no generally accepted approach to the characterisation of vision systems and none of the techniques published in the literature are suitable for the analysis of the system under consideration. The case is made for a novel approach to such evaluation, suitable for characterising the system options for a particular application.

5.2 The case for performance characterisation

Quantitative performance characterisation has not traditionally been the norm in the vision research community and is only now becoming more accepted. It is important to understand the standard approach still common and the rigorous alternatives proposed to see the need for a 'third way' in vision system performance characterisation.

5.2.1 Traditional approaches to testing in vision research

Computer Vision as a distinct research discipline can be traced back to its emergence from computer science and electrical engineering more than 30 years ago. Over this time, the primary aim of researchers has been to develop ever more effective methods to allow a system to correctly perceive details of its visual environment.

During the last 15 years, there has been a growing unease throughout the vision community over a perceived lack of a coherent system for the evaluation of the capabilities of the diverse approaches available for solving common problems.

To quote from [110]:

> *"we are willing to develop one more edge detector, but we do not want to develop objective and quantitative methods to evaluate the performance of an edge detector. About three decades of research on edge detection has produced N edge detectors without a solid basis to evaluate the performance. In most disciplines, researchers evaluate the performance of a technique by a controlled set of experiments and specify the performance in clear objective terms"*

In common with other many other fields, the pressure to publish in the vision community is intense and, as indicated above, a bias exists in favour of the conception and implementation of novel approaches to solving both new and classical vision problems.

The traditional approach to moving ahead in the field has been to follow the software implementation of an approach with sufficient testing to illustrate the validity of the concept on a small number (1 - 4 per [20]) of images. The rationale for the choices of number and of nature of the images was not commonly reported, neither was whether the results were representative of average or best performance from a larger set. At this stage, an initial paper could be published before continuing with the research endeavour by enhancing the existing approach, adding additional steps or moving on to the next concept, with the goal of writing the next paper as soon as feasible.

Also, the common practice is that results published to illustrate the quality of a method are given in the form of image-like entities. This representation is both a qualitative indicator and one which implicitly involves subjective factors in the choice of display format. Further assessment of such results is complicated by being inextricably entangled with the observer's own high-level visual processing. This processing occurs on an unconscious level and so cannot be set aside, rendering the evaluation of the results dependent on the observer, potentially different for each reader.

Even with concerns of subjective factors set aside, the use of such a result format can be misleadingly seductive where the end 'user' is not a human, but a further stage of processing or other automatic system. Such systems do not have recourse to the high level processing that is unavoidably assumed in using this results format.

A consequence of the traditional approach is the generation of many candidate solutions to each vision problem, with very little objective and comparative review of their performances. Subjective comparison is common, for example the assertion that *"Canny's edge detector is best"* is commonly made [22], but this sort of statement is not quantified, is without context

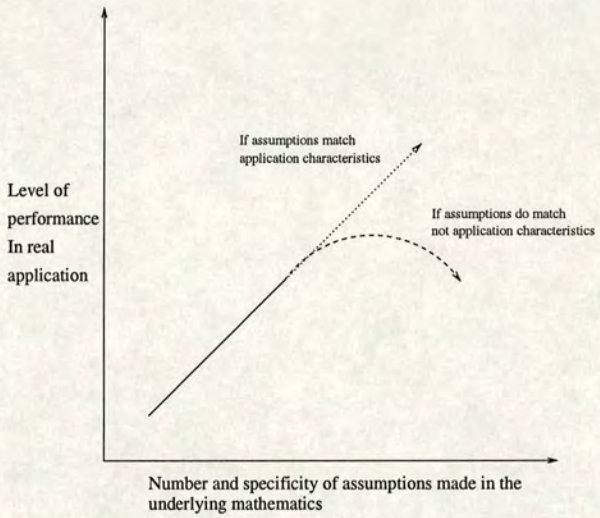

Figure 5.1: *Graphical representation of Bowyer's Conjecture for performance as a function of mathematical sophistication*

and so is essentially not open to confirmation. To quote again from [110]:

> *"Vague justifications, such as subjective evaluation of images ... should not be allowed"*

In the absence of the robust quantitative evaluation of results in the field, it is suggested [18] that it has become commonplace to measure progress by other standards. Apart from using the quality of image-like results, there is a tendency to associate the complexity of the algorithm's mathematical basis with advancement. Thus there is a trend toward ever more complex approaches without there always being a solid basis to indicate that performance is actually improving.

While greater complexity of theory **can** offer the possibility of superior performance, this will only bear fruit where the mathematical basis of the algorithm and the assumptions inherent in it are appropriate to the problem. *Bowyers Conjecture* [18] can be well illustrated graphically (Fig.5.1), showing the potential effect of increasing complexity in isolation, without an established link through empirical testing to complete the development cycle.

Figure 5.2: *The Val D'Isere cable car*

5.2.2 Performance characterisation in other fields

A useful comparison of the field with other engineering disciplines which have comparable pressures toward improving quality and reducing costs is made in [19]. A specific example related is that of the *Val D'Isere cable car* (Fig.5.2) built in 1931 and extensively repaired, refitted and redeveloped over its 70 year life to date, still transporting approximately a quarter of a million persons each year. This system's reliability and extensibility have their foundations in the rigorous demands of the civil engineering design methodology. Testing has to be sufficient to explore all foreseeable modes of operation, to chart the system's failure modes and to specify its operational parameters. Comprehensive data sheets have to be constructed on all relevant system parameters before such a system can be accepted for commercial usage, due to the safety critical nature of the design factors.

Where new failure modes are encountered (the infamous example of the Tacoma Narrows Suspension Bridge is cited), the underlying causes are identified, solutions developed and the information gained is absorbed into the discipline's body of knowledge. Similarly, in aeronautical engineering, where catastrophic failure occurs, the black box recorder is of singular importance in preserving failure data to feed back into the design process.

Electrical engineering is perhaps most closely allied to the vision field, requiring extremely complex systems which may be subject to rapid development and which must nevertheless be designed and implemented cost effectively in a commercial context. Any proposed solution in this field whose components are not extensively tested, characterised and validated with data sheets specifying all relevant properties is unlikely to be considered seriously.

Compare this with the body of work in the computer vision literature and the distinctions are clear. The legacy of the traditional approach is a large number of candidate algorithmic solutions to vision tasks, tested in a limited manner with little quantitative performance evaluation carried out upon them. There is usually no data on failure modes of systems and so neither overall performance nor range of operation can be clearly stated.

Unlike in the other fields described, there is no clear record of exactly what has been attempted to solve a certain vision problem, what did (and did not) work, how well it worked and what the effect of changing various conditions is. There is no clear basis for predicting the capabilities of systems on an arbitrary data set.

Courtney [19] provides an account of a practical instance of the problems this can cause by citing the proposed commercial development of a surveillance system incorporating a variety of detection methods, including vision-based components. In brief, a detailed data sheet was required for each component, which was not available for the vision-based items and which led to these being replaced in the final implementation. Financial resources flowed back to the non-vision developers only and anecdotal evidence suggests this is a commonly occurring pattern. The projection if this pattern continues is a dwindling of funds available to vision researchers and a consequent hindering of development in the field.

5.2.3 Objections to performance characterisation

Despite the problems caused and projected for development of the field by a lack in the experimental component of vision research as compared with the amount of theoretical work, there has been some resistance to a shift towards more empirical performance evaluation. Some 10 years after the 1985 Computer Vision Workshop, where two significant papers ([111], [112]) brought the controversy over such testing to the fore, Foerstner [113] considered in detail a variety of these objections to performance characterisation of vision solutions using empirical data. More than five years on again, evidence that the principle is gaining acceptance and interest is observable in the number of workshops and discussions around the topic in the literature. The old attitudes retain sufficient dominance however that it is worthwhile to address some of these objections here.

Task dependency

It is stated that meaningful evaluation of any visions system is task dependent and that evalu-

ation of a particular approach in one task context may not transfer to others. This is answered by recommending that the developer be responsible for specifying a characteristic set of *variable* quality measures that can be evaluated in the context of each new task. Each vision component should have its own data sheet giving performance ranges in terms of all relevant characteristics.

Complexity

Vision systems are usually comprised of many small algorithms whose interactions can be data dependent. This makes overall evaluation an intractable problem and even where data for the individual units is available, the results may not propagate to give an overall assessment. This can be mitigated with careful design: the system can be constructed from modules which may be independently evaluated and this evaluation can employ probabilistic measures which do propagate acceptably for the overall system.

Tuning parameters

Another feature of vision processes which is cited as making empirical testing undesirable is the number of *tuning parameters* commonly used. These are variables such as thresholds for evaluating the significance of differences: values which are set during initial tests to give the best results on the input data. Evaluation effort does indeed grow with combinatorial complexity with the number of free tuning parameters, but rather than being an argument against testing, this can be seen as one to minimise the number of such parameters in the design stage. They should ideally be employed only where they have a well-defined and appropriate theoretical significance which informs the setting of their values. The process of eliminating superfluous parameters can itself enhance understanding of the system.

Resources required

The final purely technical objection is the cost in man and machine hours required for extensive empirical testing, which may be beyond many research budgets. Most particularly, obtaining *control* or *ground truth* results for the test data can be very difficult and time consuming. In part, the answer here is to consider the wider perspective, viewing cost versus benefit for the field as a whole. Further investment may be required in the short term but without an improved empirical foundation, vision research will not be on an equal footing with other fields in attracting commercial interest and funding. The long term costs are likely to far exceed the short term resource increment. Another part of the solution is the creation of common publicly available data bases of test data which can be used as standards in a *benchmarking* process. Not only would the costs of data gathering be shared but this is the only way that results of testing could

be made fully comparable between research groups. A longer term goal is the standardisation of vision modules, with specifics laid down for required performance and interface characteristics.

Acceptability of empirical testing research

The last objection of all is not technical, but more psychological and sociological in nature. This is that studies of extant vision approaches, including quantitative empirical testing have tended to be less well respected than the implementation of novel algorithms in the vision research community. Further, the 'rule of thumb' time relationship of *theory: implementation: testing = 1:10:100* [113] means that research effort focused on testing gives a significant potential reduction in the rate of paper production. The increase of workshop numbers and interest in the subject is an encouraging indication that the situation may be changing in this respect.

5.3 Approaches to performance characterisation

5.3.1 Current approaches

The current approaches to performance characterisation in the computer vision literature are presented in section 2.2. Although the different approaches emerging in the field for performance characterisation are predictably varied, there are significant areas of commonality and important issues and objectives which emerge.

The importance of empirical testing

A core message of all of the current endeavours is that, to develop as a credible scientific discipline with the commercial and funding status concomitant with this, the vision field needs to complete a paradigm shift in respect of its experimental methodologies. Comprehensive tests of old and new algorithms are needed to assess their abilities on real test data.

The need to choose suitable test data

To obtain significant and useful results, the data used in testing must be both of sufficient quantity and variety to encompass the range of situations in which a candidate approach is intended to function. To promote comparability, there should ideally be a commonly available database of benchmark images to use in performing tests.

The need for test results to be quantified and comparable

Quantifiable and comparable results on the performance of vision systems are essential for

measurable and communicable progress in the field. Built upon the benchmarking database, common test and evaluation procedures are needed to facilitate direct comparison of performances and the results should be presented in a standardised manner.

Exploration of failure

The works discussed in section 2.2 all recognise that exploration of the conditions under which systems fail is of key importance, as is mapping of operational limits. The problem specifications given by users are commonly couched in terms of the maximum number of failures which are acceptable and in what conditions.

High resource demands of empirical testing

Set against the importance of obtaining empirical test results are the costs involved in the process. There is a disproportionately higher investment in terms of computer and person hours required to complete thorough testing of vision algorithms, which can potentially be mitigated by increased standardisation and the consequent sharing of costs.

5.3.2 A 'third way'

The most common approach to designing a novel system for a practical application is to take an overall problem specification and use the results available in the literature to suggest possible avenues of investigation which lead to a solution draft. This solution is generally implemented monolithically, with testing on the composite solution as a whole. As noted, the data from the literature used in this is usually only sufficient to give an indication of promising approaches.

As an alternative, current research efforts focus on performance characterisation of vision systems at the algorithm level. This is quite reasonable and important for the long term credibility of the field, taking again the comparison with electrical engineering and the characterisation of its re-usable components. Considering the effort required to characterise the performance of each algorithm in even a 'simple' vision process however, the resources required to put this into effect generally will be very great.

Although this seems to represent the only reasonable long term goal in the field, current end users need a way to make a comparative evaluation of approaches until this element of the field matures. A 'Third Way', straddling the subjective, qualitative evaluations of the past and the full comparable, objective and quantitative characterisations of the future would be a valuable tool for the present. One such approach to vision system evaluation is proposed here and its use

on the system implementations as described in Chapter 4 is illustrated.

5.4 A novel performance characterisation approach

The primary contribution of this project is the creation and evaluation of a new methodology for the integrated design and characterisation of an appropriate practical solution to a specified vision processing problem. Practical aspects of the design steps of the approach have been presented in Chapters 3 and 4 and will be discussed further in section 5.4.1. The performance characterisation approach, suitable for assisting in the selection of algorithm implementations to be used in a modular vision system, is presented in the following sections.

5.4.1 Modular design

An integral part of the approach is that object oriented principles be exploited in the design and implementation of the system. This will impose restrictions in respect of functional abstraction and interface stability. In essence this means that each module is designed so that an application designer need know nothing of its internal workings. At the system specification stage, the data objects to be passed into and out of a module are specified. This specification of an object's interface characteristics is all that is needed to ensure mutually compatible and interchangeable modular components.

In this approach, each module is considered to correspond to a distinct candidate solution which may correspond to one algorithm implementation or a fixed combination of these. The assumption is that neither full performance characterisation data on the modules nor the resources to derive such data are available. The context is where a vision-based system is required for a particular application to be run on a specified or semi-specified system. Here, the parameters of acceptable behaviour of *the system overall* will have been stated, either as specific values, acceptable ranges, or mixtures thereof.

The first step in the planning of the evaluation process is held in common with general object oriented (OO) system design principles. This is the deconstruction of the overall process into the appropriate individual modules. From the perspective of pure OO design however, module choice is dictated purely by application of the principles which dictate what constitutes a valid and efficient object. The choice of module definition for the purposes of performance analysis

however will be influenced by more diverse considerations.

Here, the module is recast as the basic indivisible unit for characterisation processes: the more modules the system is split into the greater the level of resource requirement and the detail of results. At the finest level, the modules could be taken as individual algorithms, reverting to the algorithm-wise characterisation case, at the other extreme the system would be a single monolithic entity.

The two critical factors to be balanced in deciding the module designations are how fine a level of characterisation is desired and what budget exists to perform it. The decision over this balancing is at the heart of this performance characterisation approach. The more modules that are used, the finer the results of the characterisation and the greater their potential for re-use. Reducing the number of modules used will reduce the resources required for performing the testing.

An additional consideration in some cases will be what 'off-the-shelf' modules are available to be used in the system to take advantage of the re-use principles of OO design. An advantage of this system is that the designers are free to employ any such modules without restriction to those for which performance data is available.

Similarly modules may be re-implemented based on existing work in the vision literature for which subjective, qualitative evaluations may exist. The key element of this design approach is the previously noted existence of a vast range of candidate solutions for many vision problems (here constituting our modules). The original evaluations serve as a starting point in the process of choosing promising approaches to characterise for possible use in the final system. The assumption is that there will be a sufficiently large range of solutions for the module to allow considered choice of a subset to be test-implemented for characterisation within the context of the overall system.

After the level of modularity for characterisation purposes has been fixed, and the candidate solutions for each module selected, the design of each is completed in greater detail. It is vital to include consideration of module-wise testing approaches at this stage to allow the functioning as stand-alone units to be established independently.

Each vision module is then implemented separately, either as an off-the-shelf solution, re-implementing a variant of an existing method as discussed or from an original in-house concept

where appropriate. The basic functionality of each module is tested using the 'traditional' vision research principles: test of performance on a small number of images (synthetic and real) with performance evaluated from image-like results where appropriate. The tuning parameters of each module are set during this stage to optimise the results on the limited test data.

Note the important point that elements of subjective interpretation are introduced into the evaluation approach at this stage. Similarly, with the tuning parameters set in this way, it cannot be assumed that the full range of behaviours for the system overall will be explored. These are both trade-offs needed to achieve a balance between results quality and resources required.

Relying on the functional abstraction qualities of the object oriented design, the full system should be implementable as alternate combinations of the individual modules. The number of combinations possible is:

$$\prod_{i=0}^{n} m_i \tag{5.1}$$

where:
$\prod$ signifies the product
n is the number of modules
m is the number of candidate solutions for module i

This may be reduced by fixing a certain module after preliminary results if it is known that there is significant independence between modules. A particular module combination is referred to as a *system variant* in this thesis .

5.4.2 Test data selection

Quantitative performance evaluation is carried out independently on each system variant using the same database of test data. The data should at minimum be representative of the predicted application domain and for greater generality could be extended to cover all other cases that can be anticipated. To allow comparison and repeatability of the tests, the test data must be captured in advance of the testing and stored in an appropriate format. The test data should be retained after testing and ideally made available to external parties, e.g. by posting on the web.

The exact decision on the format in which the data is stored will depend upon the specifics of the test, but several general points are important to consider. The storage format should be non-

lossy: this means that it must be possible to reconstruct the original data exactly from the stored format: any loss of accuracy or precision could bias the results. The storage format should not be prone to the creation of artifacts: this is really another aspect of the previous point, in that methods where approximations are made in the storage process may give alterations to the data are not useful. This means that storage as compressed files (single *MPEG* files or multiple *jpegs* for example) is not feasible. As it will be desirable to make the data available to external parties, most commonly on the web, the storage space required must be suitable in the context of the resources available.

Although the testing must be off-line, due to the requirements for common data and repeatability, the capture of data should be planned to incorporate the important features of the intended application. These must be evaluated as part of the planning process and stated along with any limitations and assumptions made. Where appropriate, thought should be given to evaluating the ground truth or control results during the capture phase so that any physical measurements required can be performed. In cases where ground truth is to be evaluated separately, this must be stated and planned well in advance, as this usually represents a significant draw on resources.

5.4.3 Test metrics and procedures

Once the system variants have been implemented, the database composition specified and the test data captured, the performance evaluation may be commenced. The first stage will usually be the estimation of ground truth as noted. Performance should then be evaluated using metrics appropriate to the task domain and should include evaluation of failure modes, estimates of resource requirements and analyses by data type.

The indicators will not be as quantitatively precise as those appropriate to algorithm-level characterisation and may not include information on significance levels, stochastic result formats and error propagation data. The key features which must be present are that the metrics be:

- Quantitative
- Comparative between system variants
- Sufficiently accessible for users to make an evaluation of the optimum system variant for full implementation on the final platform.

These metrics will present the basic comparative performance/resource requirement information which will then be subjected to further causal analysis. From this analysis, secondary results can be obtained in the form of hypotheses as to system variant abilities in the general case which will be the bases for the final optimality decision.

5.4.4 Assumptions and limitations

Constituting as it does a balance between information quality/generality and resource requirements, the outlined testing approach makes significant assumptions about the system and the needs of the user and has clear limitations which should be emphasised.

1. A large assumption is that the qualitative evaluation of each module and allied fixing of the tuning parameters is appropriate and will not distort results beyond usefulness. While it is clear that reducing the degrees of freedom in the system overall will inhibit the opportunities for achieving optimum performance, it is necessary to forestall the combinatorial explosion which their being left as free variables would cause. This is a key thrust of the approach: a sacrifice of optimality and accuracy to allow an interim comparative quantitative evaluation with restricted resources.

2. The interdependence of module performances is not addressed: although the intermediate results *formats* are prescribed during the design process, the *quality* of results will vary between module alternatives and subsequent modules may be able to deal with this variation to different degrees.

3. The evaluation is restricted to performance in a specific application on a particular system and so does not have the generality of algorithm-level characterisation. A mitigating factor in terms of more general usage is that useful results will be transferable between similar applications. For example, many factors relevant from a consideration of car park surveillance will be useful in evaluating options for a system used to watch for shop lifters. In terms of the system specificity, approximate mappings will be possible as long as the system specifications are included with the test results.

4. Due to the less robust nature anticipated from the results in terms of statistical significance, stochasticity and error data it will not be easy to make robust quantitative predictions as to the performance of extended/enhanced systems. If possible, where this

contingency is likely, such systems could be included as a system variant during the planning stage of the system.

Given that these limits and assumptions are taken into account in usage, this approach still offers a useful medium-term compromise solution to the lack of quantitative performance characterisation data for use in constructing vision-based applications. The results will be of great use within the group/company in developing a novel composite solution and should be useful to third parties in constructing sufficiently similar systems. Further, the contribution made to a growing distributed database for such evaluations will be of use for succeeding researchers.

The application of this proposed evaluation strategy will be illustrated on the current example, that of the vision system proposed in Chapter 4 to run on the platform described in Chapter 3. The system as proposed has alternative solutions for each of three variable modules and will illustrate the level of resource required at this level of complexity.

The specific choices made in applying the approach will be considered first, followed by the full experimental detail in the balance of this chapter. The evaluation results will follow in Chapter 6, with conclusions as to both the system choice and the evaluation approach's abilities in Chapter 7.

5.5 Applying the novel approach

The system to be evaluated is that described in detail in Chapter 4. To summarise some key points:

- The system is intended for use as a semi-automated surveillance application in its final version and is to run on the Indigo platform discussed in Chapter 3.
- System input will be colour digital images of a scene of a constant size which will be captured in real time in the final version.
- The principal outputs of the final system are to be alarm signals generated by analysis of the trajectories of pedestrian objects which have been segmented, classified and tracked in the images. This output will be augmented by the transmission of the original images for review by human operators.

- Key modules to be put under test are: background/foreground segmentation; object differentiation/labelling; object classification and object tracking.
- There are two modules to be considered as candidate solutions for each of the first three modules mentioned, in order:
  - median vs mixture of Gaussians segmentation
  - connected component vs boundary based object labelling
  - model-based vs shape based object classification

5.5.1 System specification

As noted previously, the first stages of the performance characterisation relate to the module choice criterion within the OO design process and so are discussed fully in Chapter 4. The module specification for test purposes reflects the implementation of particular vision processes both based upon variations of existing work in the field and new approaches.

The factors influencing this choice included the desire to obtain interesting results for the comparative suitability of these approaches within the context of the designated system and the need to keep required testing time within the constraints of the time available as part of the PhD project. Within the modules, maximum use was made of existing 'off-the-shelf' implementations for standard processes including those for principal component analysis, the simplex method and Gaussian elimination.

The module specification also incorporated consideration of the potential for testing as stand-alone units. Each module was supported by supplementary routines to allow the presentation of results as image-like entities which could be evaluated in the traditional manner. Again discussed in full in Chapter 4, the preliminary testing of basic functionality allowed the setting of tuning parameters for the modules based on an assessment of the quality of the image-like results.

It is valuable at this point to note an additional limitation of the performance characterisation process, not covered earlier as it is not unique to this evaluation approach. In any such empirical evaluation process, that which is tested is the performance of a *particular software implementation* of a process or algorithm. Clearly a poorly implemented version of even the best theoretical approach may well under-perform an optimal implementation of an inferior one.

In the long term, for characterisation at the algorithmic level, this difficulty may be avoided by the availability of standard implementations of vision processes. An example of a suitable model for this sort of standard database is the growing Intel Vision Library [114], formally launched in June 2000. To quote from the launch statement:

> *"We believe that the open source availability of this library will accelerate computer vision research and ultimately hasten the day when computer vision can be used in consumer products ..... Working with academia has allowed us to consolidate the best known computer vision technology and the latest research into this software library"*

As yet, the library does not offer quantitative appraisal of the implementations, but is nevertheless a significant step towards the ultimate goal.

Until this and/or other such facilities mature, it is only possible to attempt the best and most carefully considered implementation in all cases, subject to the abilities of the author. It can only remain to consider that in real applications the information of interest is indeed that of the best available implementation and so this information, though potentially deviating from an abstract ideal is nevertheless the most relevant.

As there are three variable modules, each with two alternative solutions, equation 5.1 indicates that there are eight system variants to be evaluated in this case. Each separate variant must of course use the same implementations of common module solutions and must be compiled and linked using the same options and compiler program on the same platform.

5.5.2 Scene specification

The critical choice of the test data to be used is informed by the *a priori* knowledge of the intended application, that of remote, semi-automated video surveillance. To recap, this is the process of automatically monitoring a video stream to attempt to detect 'significant events', the detection of which could be used to trigger specified alarm signals to a human operator.

These considerations provide the initial parameters to use in constructing the test database.

- The test data should be of real scenes which simulate the anticipated surveillance environment. This will include consideration of camera placement, distance from scene,

rotation, zooming and movement limits.

- The data must reflect the range of dynamic variation of the environment which is anticipated in the final application. This will include lighting variation, meteorological effects and levels of scene activity.
- An emphasis should be put on the type scene variants which earlier work in the area have qualitatively described as 'difficult'. This will allow a better exploration of the system's failure modes.
- The control exercised over the environment (precision of camera placement/orientation, internal reflection effects etc.) should not exceed that which can be reasonably assumed in the final application. Applying a disproportionate control over such effects would give an exaggerated evaluation of the system's robustness.

Before beginning construction of a new database using these parameters, a review was conducted of datasets published on the web. It was clear from this review that no database suitable for the current testing was available and so an original dataset would have to be constructed.

In the detail construction phase, consideration of these factors along with further knowledge of the problem domain allows additional assumptions and data specifications to be made which can restrict the testing resource required (although each such restriction will, as noted, limit the generality of the test results). These assumptions and data specifications are set out below, with consideration given to their limiting effects.

Automatic variation of camera parameters

A simplifying assumption required for the modules under current consideration to be appropriate is that we are dealing with fixed stationary cameras. If automatic panning, titling or zooming is assumed, the system would require further functional development to be viable.

Manual variation of camera parameters

It is assumed that in this semi-automated system the human operators have manual control over cameras and may alter the pan, tilt and zoom settings themselves, on following up an alarm signal for example. It is known that the current system will not function usefully for that camera during this adjustment process and upon concluding such observation, it is not assumed that the system will be returned precisely to its original settings.

Furthermore, the system as described in Chapter 3 allows cameras to be set up and relocated

with great facility by offering a 'plug and play' interface with the user's computer network. The placement of cameras is more likely to be carried out in-house and ideal 'professional' scene composition cannot be assumed.

The upshot of these considerations is that the system must be flexible enough to adapt to such perturbations in scene registration and that evaluating this ability should be considered in the control in the capture process. Although no attempt is made to explicitly model the perturbation process, the camera set up should be allowed to vary within the limits of manual configuration and reconfiguration.

External scenes

The primary use of surveillance systems of this type is in restricted area security applications and CCTV systems, the majority of which are focused on external scenes. Furthermore, external scenes constitute a more challenging environment to vision processes as they are subject to greater variation outside the control of the user. External scene data is used for testing then, allowing maximum exploration of the system's failure modes.

Scene occlusion

By a similar argument, data selection will be weighted toward challenging scenes containing large amounts of potential occlusion which provide opportunities for camouflage of pedestrians. The scene should cover the range from zero to full occlusion to ascertain at what point the system fails in this respect. To further task the system, a portion of the potentially occlusive elements should be subject to oscillatory variation due to environmental factors.

Scene activity levels

The scene should to be one where there is a significant throughput of pedestrians and a variety of scene activity levels over time. It should intersect with a main concourse and/or a building entry way where the focusing of pedestrian traffic will result in intense activity levels at certain times during the day. Data must also be captured for periods where activity is sparse representing, for example, the weekend period for a commercial environment. There should be a potential for non-pedestrian moving objects within the scene, to test the system's discriminatory powers.

Camera placement

There are further practical considerations to take into account in choosing the capture environment for the scene. Most external security installations involve capture by external cameras, essentially embedded in the environment. Captured images can be monitored in real time at

a central location. The capture method which would most closely simulate this would be the secure placement of a similar camera in the test environment, with long sequences transmitted to a central location for storage and the selection of interesting scenes off-line.

Without the resources for physically implementing a simulated system of this nature, the next best approximation would appear to be manual capture of images from a supervised external camera with on-board tape. The arguments against this are in respect of its unwieldiness as a practical approach: for each capture session, the camera would have to be moved to its external location and then supervised throughout the potentially lengthy capture process. A significant drawback is that this would introduce an unacceptable lead time to consider in 'reactive' capture sessions (see section 5.6.1). In order to capture the data variations described, especially the more unpredictable meteorological variations, a quick response time is needed. As the camera cannot be securely mounted in the environment, and must be setup afresh for each capture session, the interesting scene data may be missed.

It would also expose both camera and operator to meteorological conditions which, though interesting in terms of scene composition, have an adverse effect on both humans and electrical equipment. Further more, for capturing long term variations (possibly using time lapse capture) the approach is not feasible.

Accordingly, the choice was made to locate the camera indoors, in a secure local environment where it could be left unattended for long periods of time and where it could be set up for reactive capture quickly. As the camera was to observe the scene through a window, this introduced the possibility of additional artifacts due to dirt, scratches and reflections. This will complicate the system's task, but is in fact particularly appropriate for the evaluation of this application.

As noted above, the redeployment of cameras is greatly facilitated on the final platform and the likelihood of 'casual' camera placement is accordingly increased. In recognition of this fact, the decision was made to reproduce the camera placement, orientation, and zoom level by hand/eye measures only, to replicate these effects anticipated by this.

Final scene specification

Bringing together all these considerations a review was conducted of the scene/camera placement options available in the local environment with the result that the camera was located on the third floor of the Alrick Building, part of the Department of Electronics and Electrical Engineering at the Kings Buildings site of the University of Edinburgh. The camera was moun-

ted on a tripod which itself was positioned on a window ledge area and angled to observe the pedestrian concourse in front of the building. The distance and angle from which the scene was observed effectively approximates a pole-mounted or wall-mounted surveillance camera as well as the 'casual' repositioning of a mobile unit.

The details of the monitored scene which make it particularly appropriate for the surveillance testing are summarised below:

- The scene includes a much frequented section of the concourse in front of the Michael Swan Building. This building houses lecture theatres, laboratories and a restaurant and presents a varied activity profile over 24-hour and 7 day cycles. The area is a popular meeting place and so pedestrian interactions can be captured in addition to simple scene transitions.
- The scene includes two entrance points into the Michael Swan Building, one clear and one subject to severe occlusion (approximately 70% as evaluated by eye).
- A section of road in the foreground offers a variety of non-pedestrian moving objects and the introduction of new background features in the form of parked cars.
- Potential for occlusion in the scene varies from zero to total with mild occlusion epitomised by a single interrupting branch, severe occlusion at the second door (as noted above). The is also a lamp post which illustrates the special occlusion case of vertical bisection of an object and a handrail to cover horizontal bisection.
- Scene vegetation constitutes a significant oscillatory element to both background and occlusive elements.
- Artificial lighting exists within the scene (lamp post, windows, wall lights) allowing investigation of night/twilight conditions.

The endeavours to capture the widest possible range of scene variation will be a combination of proactive and reactive steps. The initial planning will encompass obtaining scene variants with predictable variations: full natural light/ twilight/ artificial light; sparse activity (Sunday capture)/ average activity/ busy (around lunchtime on a weekday). After representative sequences for these instances are obtained, further capture will continue in reaction to unpredictable novel meteorological conditions.

5.5.3 Data storage

As noted, after capture, the data must be stored in a non-lossy format, not subject to artifact generation. The design of the system has focussed on the use of sequences of uncompressed digital images in the binary *.ppm* format as these simulate the nature of data which will be available on-line on the final platform.

Accordingly, the image sequences selected from the captured data (see below) are converted to sequences of these binary *.ppm* files. The files require considerable storage resources (approximately 1Mb for a 640x480 pixel (vga) image file) and so the test data is stored on CDs prior to testing. The sequences in this format are made available on the web [115], to allow third parties to use data in the same format to replicate/extend test runs.

5.5.4 Event and ground truth specification

In conceiving the test procedures and metrics to be used, the starting point is again consideration of the application in which the final system is to be used. The projected goal is the detection of significant events in surveillance footage to allow development from an entirely manual approach to a semi-automatic one. The first issue to address is what constitutes a significant event in this context and this is an informed assumption based on knowledge of the application context.

We define a *pedestrian event* to be the behaviour of a pedestrian object within the scene over a sequence of consecutive frames. The significance of this can be evaluated independently in terms of the spacetime coordinate ascribed to that pedestrian object. It is important to decide how the 'correct' analysis of the scene in terms of these events is to be defined and evaluated in advance of system testing, to give a ground truth for the data set. Predicted results compared with actual system-derived answers are a valuable element of good experimental practice.

The decision of how to obtain the ground truth in this case again draws on consideration of the final application. The performance against which the system should be evaluated is that of the existing practical approach, which is the purely manual analysis of results. The ground truth here then will be the analysis of the sequences carried out by an 'ideal' manual observer.

The definition of such an 'ideal' observer in this context is one for whom the effects of tiredness, repetition, attention splitting and other psychological pressures are mitigated to the maximum

degree possible. The manual analysis will be conducted off-line, with images viewed and reviewed on a frame by frame basis as desired and in sittings of a length such that concentration does not fade.

From empirical investigation, sessions of no more than one hour, with gaps no less than 30 minutes between them were adopted. By recording the pedestrian events in this manner for each scene, a useful ground truth can be constructed using reasonable although not inconsiderable human resource.

It is important to note though that a further element of subjectivity is introduced here, although appropriate in the context as discussed. A possibility to eliminate or reduce this element would be conducting multiple blind evaluations of the scenes by different people, but this would increase human resource requirement proportionately.

5.5.5 Results format and test procedure

Having made a decision about how the ground truth is to be specified, it is necessary to consider an appropriate format for the recoding of its details, the results from the system tests and the comparison study between them.

Ground truth recording

The important elements to be recorded to identify the ground truth details are the *pedestrian events* defined above. The features of an event which it is important to record are its start and end frame number, and a description of the event type identifying the manual classification of the object, its trajectory and behaviour, including any interactions between objects.

To assist the coherent recording of such data a *pedestrian sequence event sheet* proforma was designed (see figure B.1) which assists these details being recorded clearly during the ground truth evaluation stage.

System tests results format

Configuration of the system variants to produce output in a form suitable for test purposes is initially planned during the design phase of the individual modules. The principal intent is for the design to allow the most efficient operation leading to the generation of results in the final platform implementation. In that full implementation, the system will need to pass results to a supplementary 'alarm module' which will use its own evaluation process and the attributes and

abilities of the CAMOS operating system (see Chapter 3) to transmit a signal along the network to the human operator.

To allow the more quantitative analysis of system performance at the event detection stage, a separate test harness is constructed to output the results in a format suitable for evaluation and comparison with the ground truth information. The general procedure chosen is a manual comparison of the test results with the data on the pedestrian sequence event sheets. This takes advantage of the human high level processing in an appropriate manner: to analyse and compare results using scene knowledge and automatic vision processing abilities. Although theoretically possible to attempt automation of such a process, this would inevitably require computer vision techniques itself, making evaluation a circular problem.

The principal output results format is visual: a 3x3 pixel square is placed in the original image at the centroid position of each identified pedestrian. This can be quickly analysed manually to ascertain if it is within the boundary of an actual pedestrian object. This visual representation is supplemented by text output to file: each pedestrian in a scene is assigned a unique identification descriptor (*UID*) on detection and in each frame this is used to flag the cumulative data on that pedestrian object.

The data recorded is as follows:

- frame number
- number of objects detected
- object centroid positions
- predicted position/scale of extant objects
- results of any attempts to match with new objects
- updated track history of matched objects
- classification results for new objects
- memory usage and timing data

This textual data can be used to resolve ambiguities in interpreting the visual output: for example where two pedestrians interact and both are tracked, has the track been maintained on

the two or has each been mistakenly identified with the other? This additional component to the analysis helps in adding a temporal dimension to what would otherwise be a collection of instantaneous evaluations.

The second important use of the textual output is to record the resources used by the system. Timing is implemented in two ways: first the *get_time()* command is used after the operation of individual modules and recorded in a series of variables within the test routine. At the completion of execution for the test run, the average times for the individual modules are output to the textual results file. In parallel with this, the Unix *time* command is used under the *Bash* shell to give data on *real time* (wall clock time start to finish), *user time* (total CPU time taken by the program) and *system time* (the CPU time spent on operating system calls executing the program).

A significant deviation between *real time* and *user time* would suggest other large jobs running in parallel with the testing (see section 5.6.3): the most relevant figure for our evaluation is the *user time*. The approach taken was to take the user time as the best indication of the time resource required overall and to allocate this time between modules in proportion to the splits given by using the *get_time()* command.

Memory usage was evaluated using the *get_mem(argc, argv)* command alongside *get_time()* for each module. For each system variant, the largest reported memory usage over all runs was used as the estimated maximum required by that variant.

Again, it is worth reiterating that the resource results are platform dependent and approximate so that only comparative results are available from this analysis, the mapping of which onto the final platform will be unavoidably inexact.

5.5.6 Test metrics

The choice of test metrics to apply must again flow from the consideration of the important factors to ascertain in the characterisation, informed by knowledge of the application.

In this instance the central performance measure will be the proportion of true positive identifications made. Analysis of scenes for surveillance purposes can be considered to be an instance of *binary hypothesis testing*: for any frame the *null hypothesis* will be that there are only background pixels/objects present, the *alternative hypothesis* that there is a pedestrian object in the

scene. In binary hypothesis testing two kinds or errors can occur: accepting the alternative hypothesis, when the null hypothesis is correct and accepting the null hypothesis when the alternative hypothesis is true. The first error is often called *Type I* error or *false positive* and the second error is usually called *Type II* error or *false negative*.

The propagation of type II or false negative errors in the system is commonly interpreted implicitly from a review of the percentage of *true positives*, the detection of a feature in a which corresponds to an actual instance of that feature occurring in the data. Within the context of the intended application, we will define a true positive as follows:

The detection in a frame of a pedestrian object whose centroid falls within the manually predicted body of the correct pedestrian object.

It is important to note that we are dealing with events at the frame level here: this approach allows us to evaluate the system on a larger quantity of data than if we were to, for example, average detections out over the life of a pedestrian event. More importantly, this approach is supported by the nature of the intended application: in security applications the eventual alarm signal should be generated by the detection of a pedestrian 'in the wrong place at the wrong time' for just one frame, and it is the ability to correctly detect this which we assess here. The system goes on to associate these frame level events (or *instances*), tracking the pedestrians to produce trajectory data: if it is chosen to use this in an application, its quality would be another logical choice for a metric.

The true positive recognition rate recorded should be analysed by scene type (table 5.1) and *instance type*. While scene type defines the overall environmental conditions, instance type characterises the nature of the particular instantaneous component of the pedestrian event. Because the system is reviewed in comparison with a manual analysis of the scene, the five instance descriptions were derived from the intuitive classifications made by a human observer as to the instantaneous situation of pedestrian. When the possibility that the pedestrian is *occluded*, *stopped*, in a *group* or *running* has been discounted the instance can, in the scenes examined, be classified (by the absence of any of these complicating attributes) as *simple*.

The analyses over instance type and scenes can be analysed graphically as simple bar graphs of the performance of the system variants. These results should then be subjected to further causal analysis to identify the failure modes of the system variants.

Following directly from the definition of a true positive is a corresponding definition for a false positive:

The detection of a pedestrian object in a frame whose centroid does not fall within the manually predicted body of the correct pedestrian object.

This covers both the cases where a real pedestrian object is detected but positioned incorrectly and where an object is incorrectly classified as a pedestrian. This is a very significant measure in the specification of this application: a key advantage of the application is that human operator attention is required only when a significant event occurs: if false alarms are frequent, response will inevitably degrade. Further, a required characteristic of the system itself is that transmission bandwidth used on the network should be minimised: this too is compromised by a high false alarm rate.

Receiver Operating Characteristic (ROC) graphs are very useful in assessing the overall behaviour and reliability of the system in terms of the two error types. The ROC graph as adapted for the current system shows the relation between the true positive percentage (100− (false negative percentage)) on the x-axis and the false positive percentage on the y-axis. The point $(100, 0)$ is the perfect classifier: it classifies all positive cases and negative cases correctly. The point $(0, 0)$ represents a classifier that predicts all cases to be negative, while the point $(100, 100)$ corresponds to a classifier that predicts every case to be positive. Point $(0, 100)$ is the classifier that is incorrect for all classifications.

An ROC curve is often plotted on such a graph as tuning parameters are varied for a single system, to ascertain its effect on the two types of error. We modify this approach to present data on true/false object recognition for each of the four system variants as a point in a subsection of the ROC graph. The ability of each variant is proportional to the distance from point $(0, 100)$, with different weightings attached to the X and Y components dependent on the importance attached to the two type of errors.

The companion metrics of key importance in evaluating the suitability of a system variant for use on the final platform relate to the resources required for operation in terms of time and memory. In the first instance these will be evaluated using maximum memory requirements and averages of the times for the actual runs on the test platform. This will provide simple quantitative data useful for comparison between the system variants in choosing the recommended optimum configuration.

The results thus gained are platform dependent and can only be approximately mapped to the final platform. More exacting results will require re-implementation on that platform which will not be feasible in all instances.

The final analysis will combine all the above results and analyses to give an overall recommendation as to the optimum system variant for the final platform. This will be discursive in nature and although it will contain quantitative elements these will necessarily be accompanied by qualitative and subjective evaluations due to the compromises made.

5.6 Experimental detail

Now that the general details of the application of the approach to the particular system have been specified, it remains only to identify the detailed actions and choices during the experimental process.

5.6.1 Data capture

Test run

The first step in the data capture was a 'test run' through the practical steps as far as data storage for a scene. In the nomenclature chosen this is scene **1a**, which does not appear in the results section as it was used to check the procedural plan only. This test run was used primarily to evaluate the options for frame size and frame rate to be used in the test process.

In the final application, the system will have access to a constant stream of image frames. Within the CAMOS environment, a new frame can be 'requested' by the system as soon as it has finished with the previous one and so the controlling factor on effective frame rate will be the system's own processing speed. The frame rate chosen for test could reflect an approximation of the anticipated/hoped for frame rate of the system.

This is again a subjective choice, but can be informed by considering the limits of the tracking algorithm: the more time elapsed between frames, the more difficult the tracking task is for a given object velocity. Tests of the tracking algorithm indicated that performance is poor for frame rates of below 2 frames per second. To maximise the number of events within a scene of standard length, this extreme acceptable value was chosen as the test frame rate.

On the final platform, the system must be able to cope with a variety of possible frame sizes subject to user choice. The assumption is made that results on a single frame size will be sufficient to allow characterisation of relative performance. The frame size should be reasonable in a surveillance context and must be compatible with the storage facilities available in the test.

The storage medium to be used is 650Mb CDs and examination of scene data captured suggests approximately five minutes to be a suitable scene duration to capture a range of interesting events while not being too taxing for a single session human evaluation. Balancing these considerations, an image size of 640x480 pixels was selected for use, at which size each frame corresponds to a binary *.ppm* file of size approximately 1MB.

Using these parameters, it was possible to store each required 5 minute scene individually on a 650Mb CD.

Capture procedure

The test data is captured using a Sony DCR-TRV900E PAL positioned as detailed in section 5.5.2 . Scene capture took place over a six month period from Summer to Winter, on the East coast of Scotland, during which period most typical weather types were encountered. As noted in section 5.5.2, the initial data capture covered environmental variants which could be easily predicted and so scenes **1b**, **1c**, **1d** and **2a** plus the time lapse sequences (see below) were captured in a planned program covering one week.

The balance of the scenes were those captured *reactively* over the balance of the six month period. Whenever an environmental condition which appeared intuitively novel occurred, the camera was set up to record a tape of footage. A summary description of the scenes in terms of day/lighting, meteorological conditions and the activity level of the scene are given in table 5.1 and example frames from each scene are given in figures 5.3 to 5.11. The scenes are each assigned a unique identifying two-character alphanumeric.

The image processing aspects of the differences between the scenes cover contrast variation (dusk/dawn c.f. full daylight), image noise (viability variations), background motion (wind effects) and degree of object interaction (activity levels).

In addition to these scenes to be used in the main testing procedures, two further scenes were captured at dusk and at dawn to test the background/foreground segmentation module in isolation. The detail of this capture is given in section 6.2.

Scene	Lighting	Visibility	Wind	Activities
1b	Full daylight	Clear	Mild	Sparse
1c	Dusk	Clear	Mild	Sparse
1d	Dawn	Clear	Mild	Sparse
1e	Full daylight	Rain, mild snow	Mild	Average
1f	Full daylight	Snow	Mild	Average
2a	Full daylight	Clear	Mild	Busy
2b	Full daylight	Clear	Strong	Busy
2c	Full daylight	Clear	Strong	Very busy
3a	Full daylight	Mild snow	Mild	Busy

Table 5.1: *Scene descriptions*

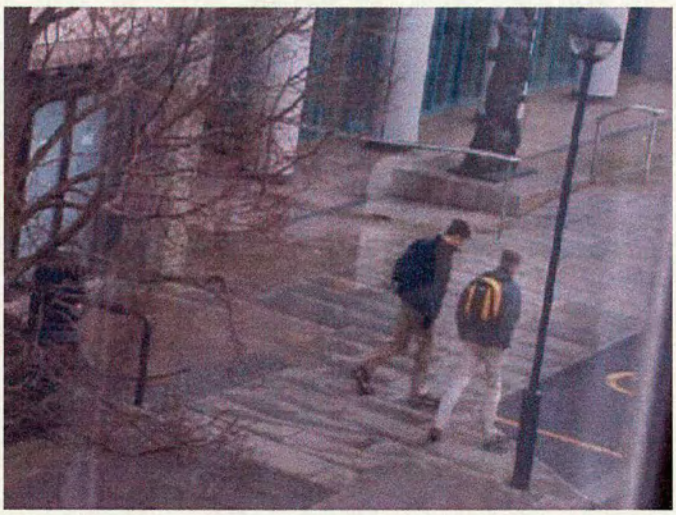

Figure 5.3: *Example frame from scene 1b*

Figure 5.4: *Example frame from scene 1c*

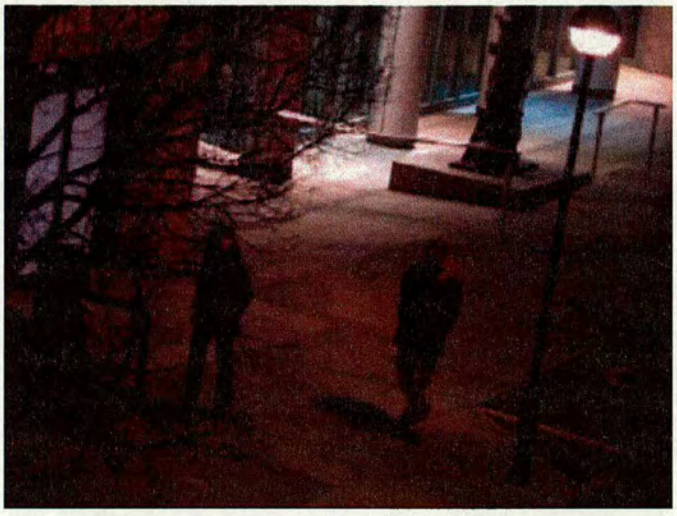

Figure 5.5: *Example frame from scene 1d*

Figure 5.6: *Example frame from scene 1e*

Figure 5.7: *Example frame from scene 1f*

Figure 5.8: *Example frame from scene 2a*

Figure 5.9: *Example frame from scene 2b*

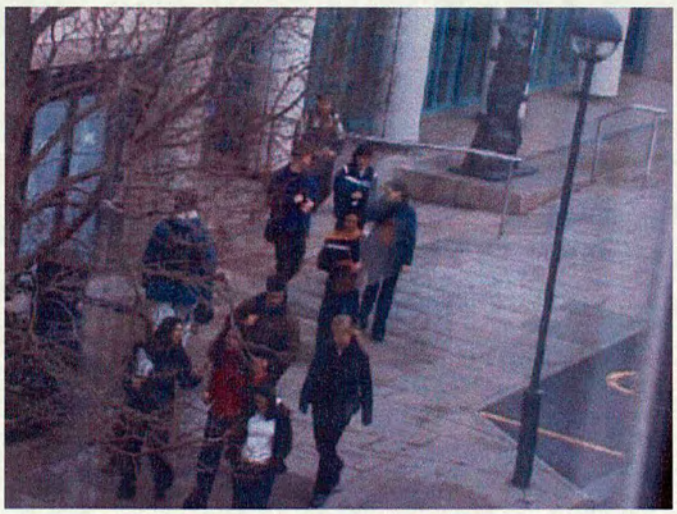

Figure 5.10: *Example frame from scene 2c*

Figure 5.11: *Example frame from scene 3a*

The detail of the capture/conversion procedure is as follows:

1. The camera is set up as described in section 5.5.2 with tripod positioning, camera orientation and zoom settings being the degrees of freedom. Recording is initiated when the scene displayed in the camera's integral LCD screen is (matched by hand/eye) a match with the desired composition .

2. The camera is left unsupervised to record scene development on a standard 60 minute *MiniDV* cassette. No restrictions are placed on the local environment, allowing other office users to vary internal illumination as desired and with the camera's automatic contrast adjustment switched on. After initial capture, the *MiniDV* cassette is stored awaiting conversion to the required format.

3. In the first stage of conversion, the camera is switched to playback mode and the full tape sequence reviewed visually on the camera's LCD screen. The initial annotations are made on the *pedestrian sequence event sheet* at this time, with events recorded by their start/finish *times* at this stage, as given by the camera's tape counter.

4. The pedestrian sequence event sheet is used to select subsequences which will be used as the five minute test sequences for evaluating system performance. The choice of subsequence relies upon the subjective human evaluation of which contain the most 'interesting' events. This involves consideration of object interactions, inclusion of non-pedestrian moving objects and the range of *instance types* (see 5.5.6) within the period.

5. The camera was connected to the a Creative DV500 x86 Family 6 Model computer with a pentium II processor and 256 Mb RAM. Connection was via the camera's DV In/Out jack using a iLINK DV connecting cable.

6. The computer uses the Pinnacle Systems miroVIDEO DVTools (version 6) software for conversion of the selected subsequence to a *.dvi* file. The software allows live capture , monitored via an on-screen window display which duplicates the camera counter reading.

7. Adobe Premier 5.1 software is used to convert from the *.dvi* file to a sequence of *.tiff* files. It is in this package where the frame size and rate may be set as desired for the output sequence. The graphical interface of the package also allows the removal of the unused sound track, to conserve memory space.

8. Paintshop Pro software is used for batch conversion of the *.tiff* files to the desired binary *.ppm* format. The frames are named using the pattern **mrSSnnnn**.ppm where**mr** denotes the author, **SS** is the scene alphanumeric and **nnnn** is the frame number from 0000 to approximately 0600.

9. Adaptec EasyCD Creator version 4.01 is used to transfer the *.ppm* files to storage CDs, one for each subsequence. The CDs are marked with the sequence alphanumeric, the textual description of the scene environment (as given in table 5.1), the frame rate, the frame size, the range of frame names and the individual file size.

5.6.2 Ground truth evaluation

The initial work for the ground truth evaluation was laid out during the conversion process, with the pedestrian sequence event sheets partially completed, to the stage where all events were listed with their relative timings. These sheets are now updated to include the frame number referencing for the start/end of each event, for any relevant interaction and for the times where behaviour type changes (e.g. stopping, running etc.).

This is a slow, frame-by-frame process: the stored scenes are loaded using an image viewer (both XV and Paintshop were used for this purpose) and, using the event sheet as a guide, frame-wise reference points manually ascertained. Each event is also allocated a unique reference number to assist in identification.

The end result of the process is a complete set of pedestrian sequence event sheets which identify pedestrian events by the range of frames in which they occur and frame-wise references for significant changes therein. This is assumed to represent the 'ideal quality' of human interpretation of the data, as each frame can be reviewed and checked repeatedly to ensure the accuracy and precision of results. There is no pressure to perform in real time and events can be interpreted based both on past and future developments.

5.6.3 Test runs

The initial phase of test runs was designed to evaluate four of the total eight system variants outlined, the subset which used the connected component algorithm approach to object labelling. This was the approach used during the bulk of preliminary testing and was superseded sub-

sequently due to the excessive time resources it required. Referring to section 4.5, it is known that altering this object growing method gives no change in results in terms of performance and so cannot be excluded in the initial analyses. The effect of this change on resource requirement is substantial and is addressed in section 5.6.5.

Each system variant is to be run on each scene, giving an initial total of (4x9 =) 36 normal performance evaluation runs plus (2x2 =) 4 time-lapse analyses of background module performance: a total of 40 runs. Each run is to be processed individually on an assigned workstation and each generates approximately 0.7Gb of results data. To keep the resource usage to an acceptable level the tests were split into four so that 10 workstations and a maximum of approximate 7 Gb would be needed at any one time.

As discussed, the test platform is the Sun Microsystems Ultra 10: in the test environment the workstations available are part of the Department of Electronics and Electrical Engineering network and as such can have jobs allocated by other users at any point during a test run. This would not affect performance and is corrected for in interpreting overall time resource usage by considering the *user time* as the relevant measure. However, distortion in analysing individual modules time resource demands (using the *get_time()* command) could be introduced by intermittent remote access.

Accordingly, a discrete window was specified for performing the test runs, specifically overnight during a two week holiday period. The network contains over 100 workstations in total and is configured so that users logging on remotely are directed to workstations with low or zero current usage. Given this, the chances of 'multi-user' distortion are acceptably low.

The test harness requires as input the sequence of frames corresponding to the scene, conditional compilation flags to indicate which system variant is to be employed and detail of the file and directory where the visual and textual output is to be sent. Compilation flags are not set to optimise the code, but configured allow debugging: if time permitted, subsequent runs with compilation-optimised code would be recommended.

Copies of the required sequences of frames of frames were copied from CD into the requisite directories and further directories were constructed for storing the output results files. Workstation job statistics were monitored approximately once an hour as an additional safeguard. At the completion of each individual run, the image-like results were themselves stored on CD ready for analysis, to minimise ongoing memory needs.

5.6.4 Comparison with ground truth

The final stage of the result derivation is the comparison of the results of the test runs with the ground truth predictions made previously. The metrics used are primarily *true positive* and *false positive* numbers as defined in section 5.5.6. In general, the process involves the manual comparison of the image-like results generated in the test runs with the the pedestrian sequence event sheets, supported by the textual results to resolve any ambiguities.

The pedestrian sequence event sheets provide the context for each pedestrian event, specifying the number of frames over which the system should have detected the object and its interactions. Manual review of the scene for that range of frames (again using an image viewer) provide data on whether a pedestrian was detected in all these frames in the correct position. A subjective measure of 'correct position' as being within the visually ascertained boundary of the person is used. The textual output is used if where any ambiguity remains as to whether the person tracked over time is the one as identified in the current image.

This is the most involved and time consuming manual stage of the evaluation process. To facilitate the accurate recording of results, a spreadsheet was developed to record performance by system variant and by event. The Star Office package, freely available on Unix, was used, facilitating direct entry while viewing images on the same workstation on which the sheet was active. A standard format within the sheet was used for each run and a 2 dimensional layout with *system variants* arranged vertically and *scenes* arranged horizontally was employed to assist comparison between them.

The general format is that for every run of a system variant on a scene, the sheet analyses each Numbered) pedestrian event individually by row. For each such event, entries are made for true positives, false negatives and a total for false positives. These entries are analysed by instance type (see section 5.5.6) and are entered directly during the analysis process.

Simultaneously with this entry process, the sheet calculates check totals to ensure that entries are consistent within and between runs. Subsequently, the sheet can be used to calculate true and false positives as percentages, to present the results analysed by scene type and instance type and to facilitate the production of graphical representations to aid analysis.

5.6.5 Resource requirements

An additional section within the sheet was used to record the basic data for memory and time resource usage for each run. Again, the spreadsheet was used subsequently to assist calculation of the maximum and average resource requirements by system variant and module. The testing was extended to cover all eight system variants. As only resource requirements were under study, the visual output option was switched off which considerably reduces the demand on the system for result storage. It should be emphasised that the time taken for both visual output and textual output was not included in the timing results for any of the test runs. These outputs will not be required in the final platform application, as alarming will simply trigger redirection of the extant live video stream.

5.6.6 Results analysis

After the raw results are compiled within the spreadsheet, the derivation of the performance results is essentially complete. The results so derived are presented in Chapter 6, including the graphical representations created within the spreadsheet.

After initial review of the results, a further stage of manual examination of the data was required to investigate unexpected trends and behaviours which manifested. This generally took the form of an instance by instance analysis of unanticipated failure modes in the system to enable more meaningful results to be extrapolated.

5.6.7 Posting of test data

After completion of the review and analysis stages, the test sequences in their *.ppm* format were posted for general access on the web. They can be freely obtained at www.ed.ac.uk/ mer/project/.

5.7 Summary

A performance characterisation approach suitable for characterising the system options for a particular application has been described. The reasoning underlying the choice of methodology has been discussed and the application of the approach to the current evaluation problem has been detailed. The next chapter will present the results of this evaluation and the conclusions

which can be drawn from them.

Chapter 6
Results

6.1 Introduction

The performance characterisation of the modular system's variants generate complicated results which are best considered using a suite of related metrics. The behaviours of the two background approximation approaches, as stand-alone procedures, are first examined using a common approach from computer vision, the analysis of output as an image-like entity.

The system's ability to perform the surveillance task of correctly locating and tracking pedestrian objects is quantitatively analysed over multiple instance types and widely varied scene conditions. The overall performance is evaluated with balancing consideration of the resource requirements necessary to achieve this performance.

6.2 Background representation: light variation

As a separate exercise from the performance characterisation of the overall system, the background generation module can be examined in terms of its ability to cope with 'difficult' sequences. As noted previously, the ability of a background estimation approach to handle lighting variations in an effective manner is a prerequisite for use in a robust surveillance application.

The modular design of the overall system was conceived and developed in such a way as to allow the operation of this unit in isolation from the subsequent object extraction, classification and tracking stages. Instead of providing input to the object segmentation routine, this module's output is redirected to a specially written routine which generates a *.ppm* image suitable for display using a standard image viewer. This was important in testing sequences of sufficient length to cover the period from full natural lighting, over dusk, to a scene illuminated purely by artificial means (and the reverse process at dawn).

Using the time lapse recording facility of the Sony DCR-TRV900E PAL camera it was possible to store fifteen hours of scene evolution by capturing two second shots every 30 seconds. Using

the procedures detailed in Section 5.6.1 this was stored on a 60 minute *MiniDV* cassette and the two key dusk and dawn sequences of 5 hours each were each converted to an *.avi* file and then to a sequence of 600 *.ppm* files. Thus the test sequences sample the original footage at a rate of 1 frame for every 30 seconds.

Using this time lapse approach can make the task presented to the background construction routine more difficult, as each new frame corresponds to an exaggerated change in the average illumination of the picture. The ability of the method to cope with more rapid lighting changes (for example caused by passage of discrete clouds over the sun on a bright day) is thus tested, as well as its ability to cope with the extreme illumination changes due to the fall and retreat of night.

As noted, unlike the performance characterisation results to follow, those presented in this evaluation of the background generation module are image-like in nature, illustrating the development of the background representation over time. Evaluation of the quality of the background representation in isolation involves analysis of these results using the 'black box' of human high-level image processing.

Conclusions will be qualitative in nature and inherently anthropocentric, which is not necessarily appropriate for evaluation of what is to be an intermediate result in the proposed application. The important characteristics of the background representation are those which dictate its suitability as input to a subsequent automated process, which may be quite different to those which allow a convincing human perception of the scene. Consideration of the precision (or lack thereof) available from results in this format will serve as a good practical illustration to emphasise the importance of the more quantitative results presented in subsequent sections.

The results presented in figures 6.1 to 6.4 show the image like representations of the backgrounds generated by the median background approach and the mixture of Gaussians background approach on the two sequences (through dusk and through dawn). These representative images are taken in most instances at regular 30 minute intervals to compare the evolution of the two methods over time.

The exceptions to this rule are the final three images for the dusk sequence which represent 30 second gaps to illustrate the performance of the candidate algorithms in respect of an addition to the stationary components of the scene (a parked car).

6.2.1 Median background

In this approach the image-like representation is derived by plotting the current background value (the weighted median average) at each pixel location directly as the value of the background image. The representation (figures 6.1 and 6.2) is convincingly realistic to the human eye: the high level processing of the brain recognises it as a good depiction of the scene's background as it varies over time.

Qualitatively, this background approach can be seen to perform well overall, adapting to reflect the change in ambient lighting and incorporating the different distribution of lighting patterns caused by the advent of artificial illumination.

A point that is well illustrated by the inclusion of the parking car event in figure 6.1 is that there is a significant latency in the approach. The calibration of the median update must allow the detection of pedestrians at slow walking speed, rather than their immediate incorporation into the scene, which would prevent reliable detection.

The consequence of this is that it takes of the order of 20 frames for the car to be substantially registered and the order of 40 frames for complete incorporation into the scene background. In a real-time application, the actual time difference this corresponds to will depend on the frame rate at which the system runs.

6.2.2 Mixture of Gaussians background

Translating this approach to an image-like representation is a more involved process, as there is no single value associated with each pixel position in the scene. A meaningful image-like representation of the background must include the derivation of a single value which best portrays the Gaussian distribution most confidently classified as background at that point. The derivation of an image-like representation of the mixture of Gaussians background is described in section 4.4.5

The representation (figures 6.3 and 6.4) is less visually compelling for a human observer than that of the median background approach. There is an 'unrealistic' specularity in some areas where neighbouring pixel values differ more discontinuously. This corresponds to the more complex probabilistic nature of the underlying process, where it is more likely that different decisions as to whether two neighbouring distributions should be updated or replaced may be

Figure 6.1: *Median background in fading light (dusk), with time incremented along the rows.*

Figure 6.2: *Median background in growing light (dawn), with time incremented along the rows.*

made.

Without more extensive analysis the prima facie conclusion, based solely on image evidence by a human observer, would be that this background representation is inferior to that given by the median background approach. Although the mixture of Gaussians approach adapts to reflect the change in ambient lighting and distribution patterns, as does the other approach, the divergence from a 'realistic' image-like representation introduces a negative bias into the evaluative process.

The inclusion of the car parking situation in figure 6.3 mitigates this initial conclusion in that there is a significant reduction in latency in this approach. By having multiple Gaussian distributions to represent recurring background values at each point, the new object is incorporated immediately into the representation as a foreground object, with no change required to the distribution representing background at that time.

From section 4.4.4 it can be seen that incorrect classification of a slowly moving object as background requires that the distribution to which it is matched have both a high weighting (corresponding to a large amount of evidence) and a low variance. As the latter requirement is not generally satisfied for even very slow objects, the calibration of the method in this respect can be skewed to allow faster incorporation of new stationary objects.

The consequence of this is that it takes of the order of only 5 frames for the car to be registered and this is immediately complete incorporation into the picture as a result of the binary nature of the background/foreground decision.

6.3 System performance

6.3.1 Performance evaluation

Chapter 5 discusses in detail the rationale behind the choices of scene type, classification of events, manual scene analysis, prediction of results and analysis options.

To recapitulate some important points:

- A *scene* is defined as a sequence of images gathered consecutively containing a collection of pedestrian events.

Figure 6.3: *Gaussian background in fading light (dusk), with time incremented along the rows.*

Figure 6.4: *Gaussian background in growing light (dawn), with time incremented along the rows.*

- A *pedestrian event* is defined as the behaviour over time of a pedestrian object occurring in a series of consecutive frames within a scene.
- An *instance* is the current image representation of a pedestrian in a frame, a series over time of which then comprise a pedestrian event.
- An instance may be classified as *simple*, *occluded*, *stopped*, *group* or *running* (See table 6.1).
- A *system variant* is a specified combination of modules comprising a candidate pedestrian tracking solution.
- A *true positive* result is the location by a system version of a pedestrian object within its manually identified boundary, a false negative is the failure to achieve this.
- A *false positive* result is the location by a system version of a pedestrian object outside of the manually identified boundary of a pedestrian.

System variants are differentiated by the alternatives employed for the three primary modules under consideration: *background/foreground segmentation*, *object growing* and *object classification*. The performance results are only affected by changing the first and last of these modules: altering the object growing method (the mapping of discrete object connectivity) gives no change in results in terms of performance and so is not included in the analyses in this section. The effect of this change on resource requirement is substantial and is addressed in section 6.4.

The four system variants investigated here then are *Median/Shape*(**M/S**), *Median/Model*(**M/M**), *Gauss/Shape* (**G/S**) and *Gauss/Model*(**G/M**).

The performance of the system versions is quantified in terms of percentage of true positive results and the comparison of percentage of false positive results. The analysis of these performance measures are by instance type and by scene. For clear analysis of the results, it is necessary to decouple the effects of instance type and scene type to be considered separately.

Instance type characterises the nature of a particular instantaneous component of a pedestrian event and can be associated with the five classifications listed above. Because the system is reviewed in comparison with a manual analysis of the scene, the five instance descriptions were derived from the intuitive classifications made by a human observer as to the instantaneous situation of pedestrian. When the possibility that the pedestrian is *occluded*, *stopped*, in a

Scene	Simple	Occluded	Stopped	Group	Running	TOTAL
1b	41	113	0	0	3	157
1c	13	75	27	0	0	115
1d	11	123	21	0	0	155
1e	51	164	24	0	7	246
1f	11	126	42	0	0	179
2a	43	18	3	100	0	164
2b	81	262	29	139	3	514
2c	112	205	7	139	7	470
3a	0	78	0	0	0	78
TOTAL	363	1164	153	378	20	2109

Table 6.1: *Numbers of instances analysed by instance type and by scene*

group or *running* has been discounted the instance can, in the scenes examined, be classified (by the absence of any of these complicating attributes) as *simple*.

It is necessary to in some way allow for classification of instances which are a combination of these relatively arbitrary instance types, for example a pedestrian may be occluded **and** simultaneously be running. Again recourse is made to an intuitive human order of precedence, recognising first *running*, then *stopped* behaviours as gross changes in pedestrian motion then aggregation into a *group*, and finally *occlusion*.

The ordering of occlusion is perhaps most open to debate, the rationale for its placement being two fold. Occlusion can describe a wide range of degrees of object loss: at its mildest the occlusion may be virtually undetectable, at its most severe the object itself may be virtually undetectable. Where occlusion is sufficiently mild to allow recognition that a pedestrian is running, stopped or in a group then it is reasonable to consider that occlusion is the secondary characteristic.

It should be emphasised that the order of precedence thus presented is relatively subjective but that the alternative to some such subjective ordering would be to separately analyse each possible combination of classes. While certainly possible and an option for future work, these combination situations were not sufficiently numerous to warrant the additional work at this time.

Scene type applies to all pedestrian events and their constituent instances in a particular scene. The classifications are again subjective and based upon the intuitive classification of a scene by

Instance type	Median/ Shape	Median/ Model	Gaussian/ Shape	Gaussian/ Model	Average performance
Simple	66.94	82.42	73.13	79.94	75.61
Occluded	33.28	45.48	52.36	60.00	47.78
Stopped	26.48	33.20	67.94	41.75	42.34
Group	72.47	69.10	62.92	65.14	67.41
Running	45.00	54.55	70.00	82.51	63.01
TOTAL	41.68	52.35	58.71	63.31	54.01

Table 6.2: *Percentage true positive identifications analysed by instance type and system variant*

time of day/lighting, meteorological conditions and the activity level of the scene as detailed in table 5.1. To be suitable for practical use as a robust semi-automated surveillance solution, a system must be able to perform adequately well across the range of likely scene condition types.

6.3.2 Analysis of true positive recognition percentage by instance type

Table 6.1 presents the numbers of instances analysed by instance type and by scene. Table 6.2 gives the percentage true positive identifications analysed by instance type and system variant, which are displayed graphically in figures 6.5 and 6.6.

Simple Instances

As predicted, the best results for true positive recognition are for simple instances, pedestrians moving individually at walking pace under no occlusion. With an average recognition rate of 75.61% though, the results are lower than would be required from a practical surveillance application. More detailed review of the instances where recognition failure occurred reveal two primary causes.

First, in several instances the size of the pedestrian object is not great enough to pass the initial thresholding to be classified as a valid object as opposed to noise. This step is common to all system versions and threshold level is set at an absolute pixel value in the current implementation, the least sophisticated approach possible. Two possible improvements would be to reduce the absolute value and/or to make it variable in proportion to the image size, giving greater flexibility and a more robust treatment of image scale variation. These changes would potentially allow smaller objects to be classified as possibly valid but would thus create an increased

load on the object classification routine in filtering out non-pedestrian data: this is another good example of the trade-off between performance abilities and resource utilisation.

The second main situation where failure was observed was in incidences where the contrast between the foreground and background were insufficient to generate a valid object at the segmentation stage. Again, this is a problem common to both approaches and is one which can apply to any segmentation approach based solely on pixel value data. Solutions to this camouflaging problem are more challenging and include the use of data fusion techniques [109] (to combine disparate data sources/processing to correct for weaknesses of any individual component) and directed enhancements to current techniques [51].

Comparing the performance of the different system versions for simple instances (figure 6.5), there is an unexpected deviation from the predicted pattern. Qualitative predictions suggest that median/shape combination would be least effective, the Gaussian/model most effective, with the other two approaches of intermediate efficacy. What we see here is better performance from the median/model version than any other, most significantly better performance than the Gaussian/model version.

In these most simple instances, it would be reasonable to predict less benefit from the Gaussian segmentation approach: greater improvements in performance would be expected in the occluded instances. However some benefit would be expected from incidences where there is a multi-modal background (e.g. background vegetation motion) and even in the absence of this, there is no prima facie reason for the result from the Gaussian approach to under-perform as compared with the median version.

Further detailed review of the failure instances in the Gaussian versions suggest an explanation: these instances occur primarily in situations where multiple pedestrians follow quickly (within 10 frames) on paths which are similar (with more than about 50% pixels in common) to each other. Considering the operation of the Gaussian segmentation algorithm, a pixel is classified as background where the current matched distribution has a low variance and a high amount of evidence. In the situation described, in the 'wake' of the common trajectory, a distribution previously matched to will still have a relatively high level of evidence associated with it. If the pixel value of pedestrians passing along this trajectory are similar enough, then the variance can in turn be low enough such that an erroneous background assignment is made, causing a false negative result.

In the current implementation, although colour images are used, they are converted to grey-scale equivalents during preprocessing: if full colour processing was adopted, the similarity of successive pedestrians would be reduced, increasing the variance in the above situation. This would reduce the likelihood of a failure due to this combination of effects.

Given this explanation of the unexpected reduction of efficiency of Gaussian segmentation, our revised prediction is that the median/shape approach would outperform the Gaussian/shape version, which is evidently not the case (66.94% for the former compared with 73.13% for the latter). Examining the instances where the median/shape version uniquely fails, the conspicuous characteristic of these is the relatively high proportion of the segmented foreground object (in the region of 20%) which is comprised from the object's shadow along the ground.

In the mixture of Gaussians segmentation approach, *illuminated ground* and *shadowed ground* can be learned as two valid background values and so substantial inclusion into the foreground object need not occur at all. The model-based object classification approach is able to cope with such erroneous inclusion quite well as only a relatively small proportion (in the region of 10%) of the outline as used in the matching process is likely to be distorted.

It is in the median/shape variant where the problem is not mitigated by either of these considerations. The shadow is included as noted and this can cause a large distortion of the registered object shape, giving rise to a reduced detection rate in this instance.

These factors taken together constitute a useful causal analysis of the anomalous performance distribution.

Occluded Instances

At 47.78%, the average recognition rate is well below that for simple instances, as predicted. As mentioned, occlusion ranges from mild (e.g. a leaf covering a pedestrian object's elbow) to severe (e.g total bisection by a bar). Without conducting a detailed quantitative analysis of occlusion percentage, routine examination of failing instances provided the expected qualitative result that the failures were occurring in proportion to the degree of occlusion.

Comparing the performances of the system versions (figure 6.5), the pattern of Gaussian segmentation outperforming the median approach and model-based classification working better than shape-based was observed as predicted according to the rationale below.

For the median segmentation approach, both obscured and unobscured points must be compared

with a weighted average of the values observed over time at that point. The value will vary according to the frequency of oscillation, and the relative values of the alternating background image objects. The results of the segmentation of this are chaotic and essentially unpredictable in the general case.

The Gaussian segmentation approach allows for multiple values to be identified as background at each point, irrespective of whether the value corresponds to the colloquial meaning of background or to an oscillating potentially occlusive object (e.g. a tree branch). Where an entirely new object such as a pedestrian moves into this scene area, then for those points not currently obscured, a new distribution will be initiated, distinct from those corresponding to background. The currently obscured points will be identified with the appropriate background distributions. This means that for a general point subject to oscillatory occlusion, the percentage occlusion will be minimised by the Gaussian approach, giving a greater probability that a sufficiently large connected object will be segmented to register as a potential pedestrian.

Comparing the object classification approaches, the model based method need only be able to make a match at sufficient outlying edge points to provide enough data for a determination of the fitness of a model match. The shape based method requires not only that a sufficient percentage of the object be unobscured and connected, but that the distribution of points be sufficiently well preserved to maintain the approximate shape.

Stopped Instances

In manual observation of the sequences it was apparent that, to a human observer, the classification of a stopped pedestrian and recognition of such in a scene will operate quite differently from the perceptual model upon which the tested system is based. With the higher level processing available to the least experienced human observer, when a pedestrian stops within a scene, even if they remain entirely motionless, they will still be recognised as alternative methods of still image segmentation are unconsciously brought into play.

All the system variants start from the basic assumption that only a moving object need be detected and anything which stops for an extended time should be incorporated into the background. It is therefore the *correct* functioning of the system to not detect truly motionless objects after a latency period during which they would become features of background, as in the example of a parked car (section 6.2).

For stopped pedestrians however, the situation is somewhat different: they are unlikely to be

truly motionless being subject to small scale motions of the arms, head upper body and at finer scales hands and face. Closer examination of their behaviour during the manual observation observation phase made it evident that normal behaviour for a stopped pedestrian also includes more gross body movements. Oscillatory motion about an average 'station' point occurs, which opens the possibility for the system variants to continue to segment out some kind of object. some areas of such a stopped pedestrian will coincide between consecutive frames and the nature of the object detected will vary dependent both on degree of motion/overlap and the system variant employed.

On examining the performance for this type of instance, the average recognition rate is, as predicted, lower at 42.34% but nevertheless higher than prima facie expectations indicated. Comparing the performances of the system versions (figure 6.5), we see the normal steady increase of performance with the conspicuous exception of the Gaussian/shape combination. This can be explained best by considering the drawbacks of the two modules not employed in its operation.

In the median segmentation approach, areas occupied relatively consistently over the 'stopped' period will be partially absorbed into the background after a latency period, as discussed. By the nature of the oscillatory motion, on average perpendicular to the pedestrian's principal axis, the points where this occurs most will be clustered about that axis. This will in effect constitute a vertical bisection of the potential pedestrian object, giving the same affect as severe occlusion, which confounds both object classification approaches.

Considering the Gaussian segmentation, the dual criterion of low variance and high evidence must be satisfied for a point to be classified as background. In the case of the oscillating pedestrian, some of the points which were sufficiently absorbed into background in the median approach to approximate occlusion do not have a low enough variance to be classified as background by the Gaussian approach. The number of effectively occlusive points and so the severity of occlusion is correspondingly reduced, although there will still be a bias towards the bisecting occlusion for the reasons discussed.

In most occlusive situations, the model-based classification approach performs best, but further examination of the failure in detail showed that the bisection case is a notable exception. The hypothesised reason is that in this situation, the system detects two discrete objects to either side of the bisection (unlike in the general occlusive case where the expectation is that those points

are located are still likely to be connected). The key difference in the segmentation results is that the objects resulting from the Gaussian approach will be larger than those from the median, although one or both may be below threshold size altogether.

The model-based classification approach tries to fit a full model to each half object (unlike in the general occlusive case where a single model fits well to the remaining boundary points of a single part-object) with poor results. The shape based approach simply compares the vertical/horizontal ratio of the objects to the threshold value and using this criterion performs relatively well.

A corollary of this consideration is that problems could occur where both objects exceed threshold size in that there would be double counting of pedestrian objects, i.e. an increase in the false positive rate.

Group Instances

At 67.41%, the average recognition rate is comparable with the high value for simple incidences and considerably above the average of 54.01%. This instance type has the most pronounced overall deviation from the average performance profile (figure 6.6), with the clearest difference being that Gaussian segmentation is distinctly less effective than the median approach. This is at first sight a most unexpected result, but can be explained as a side effect of the higher discriminating ability of the Gaussian segmentation coupled with the deliberately anthropocentric treatment of group results.

A group is perceived as the essential moving unit when a collection of pedestrians move together simultaneously with the same or similar relative velocity, often mutually obscuring each other in their progress through the frame. Rather than interpret them as a multiple dynamically interacting individuals, the human observer intuitively classifies the significant discrete entity as the group itself. In the manual observation stage then, the result was considered in terms such as:

> *"group 3 entered from the right hand door beginning in frame A, exited at left of scene ending in frame B"*

as opposed to:

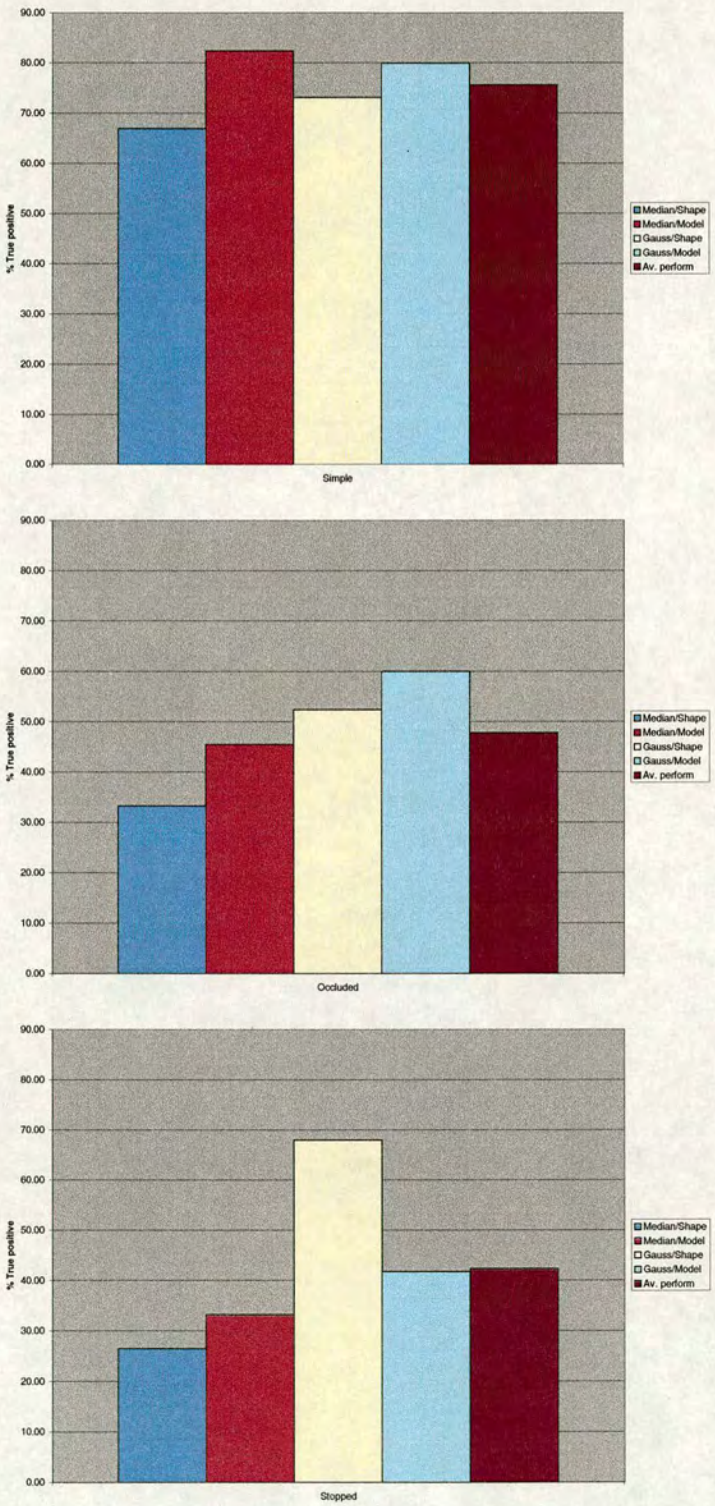

Figure 6.5: *True positive identifications for simple, occluded and stopped instances*

> *"person 1 entered from the right hand door in frame A, occluded person 3 from frame C to D, was occluded by person 4 in frame E, exited at left of scene in frame B; person 2 entered from the right hand door in frame....etc."*

This degree of inherent high-level processing is not available to the system variants under test where the individual pedestrian is the essentially unchangeable unit of detection and mutual occlusion between such individuals constitutes a significant complication to this task. In the present context of a surveillance application being evaluated against manual scene analysis it was decided to retain the psychologically motivated group classification and to state a correct identification more broadly in these instances. This approach is similar to that taken in [80].

A true positive identification is defined as one or more of the individuals in the constituent group being correctly located and tracked. This is reasonable in a semi-automated surveillance context as it is the minimum requirement for correct notification of an operator in a group situation. Given this context, the deviation from the normal pattern is easier to account for and indeed should be expected as the problem under consideration has varied.

Considering the Gaussian approach first, the previously evidenced superior performance holds, with the segmentation approach yielding the correct result for the given data: a group is registered as multiple instances of partially occluded discrete objects. The occlusion tends to be quite severe as the group mills about and such occluded individuals fail the size thresholding. These not being registered as valid objects at all accounts for the relatively poor acquisition rate for the approach as a whole. For those objects which pass the threshold, the superior ability of the model-based approach to recognise partially occluded pedestrians is reflected in the results.

In the median approach, the synchronised passage of multiple synchronised objects, if of similar grey-level attributes can be *incorrectly* interpreted as the passage of a larger merged object. These 'false' group objects have a higher overall detection rate than that of the 'true' mutually occlusive pedestrians in the Gaussian approach. This gives an apparently higher good detection rate within the definitions of the test for the median approach overall.

When the object classification stage is reached, the shape-based approach again requires only a shape ratio above threshold which is achieved rather well by the severely occluding merged pedestrians. The model-based approach is seeking to match with the normal case of an individual pedestrian for which the merged objects are a less good fit, underlying its relatively poor performance.

Running Instances

In a visual analysis, running pedestrians differ from walking pedestrians in two main respects: the speed of their transition through the frame and the nature of the variations of their body outline.

The effect of the speed of passage of a pedestrian on the background update has been mentioned when considering 'stopped' instances, the important feature being that if the passage is sufficiently slow, the pedestrian may be partially incorporated into the background and not correctly detected. At the extreme of very rapid passage however, no such problem occurs in segmenting the moving object from the scene.

Similarly, in the object classification stage, the speed of the object should present no problem: at this stage the object has been located and classification takes place independent of its previous location.

The component which may be confounded by a fast moving object is one not under comparative test in the current scheme, the tracking module. Each system variant uses a Kalman filter-based tracking module (section 4.6.2.9), optimisation or enhancement of which could improve the systems' overall performance for the running instance, but which is outside the scope of the current work.

It is in the gait variation of running pedestrians compared with that of those walking that the variation in system response originates. A running person's limbs reach extrema from the body's principal axis that are more pronounced than those of a walking pedestrian and outline variations are correspondingly greater. While not significant in the segmentation step, this can present problems for both object classification approaches.

If the the height/width shape ratio alters sufficiently as a result of this laterally expanded outline, the object will not be classified as a pedestrian. Similarly, if the distortion parameters required to match the walking model to the outline are too great, the classification will again fail.

From the experimental results in figure 6.6, the performance range (from 45.00% for median shape to 82.51% for Gaussian/model) is the largest of any instance type and is observed for changes in classification approach *and* segmentation approach.

From this it can be concluded that the flexibility of the model-based approach is greater in this area than that of the shape-based method, which is contrary to the qualitative expectations

which anticipated the model-based approach to have more problems. Changing the threshold ratio for shape acceptance could mitigate this, but at the cost of more false positives across all instance types.

The question remains as to why there is a significant increase in performance between the median and Gaussian segmentation approaches. The answer is two fold: detailed analysis of the trajectories taken by runners revealed that 56% also involved some degree of occlusion, not reported in the general results due to the precedence decision as previously detailed (section 6.3.1). As discussed in the section addressing occluded instances, the Gaussian approach performs significantly better in such situations. Secondly, the sample size for running instances is relatively small at 20, which can account for both the apparent size in the variation in performance between the two segmentation techniques and the exaggerated range of performance for this instance type overall.

Overall Performance

Considering the performance over all instance types (figure 6.6), we observe the qualitative prediction of a stepwise improvement over system variants is confirmed. An interesting quantitative result is that the performance enhancement on the median/shape variant (at 41.68%) is by 10.67% by enhancing the classification approach, but by 17.03% by focusing on the segmentation method. The fact that improving both modules will only return a 21.63% improvement in performance is a good illustration of the non-additive nature of such enhancements. The processes cannot be considered to be independent, being components of a serial process where the output of one directly effects the input of the next, the performance of which can accordingly be altered.

As was noted earlier, the overall system performance could be increased above its current modest level by using additional modules which could use further scene knowledge to enhance performance, but such strategies are outside the scope of the current work.

6.3.3 Analysis of true positive recognition percentage by scene

Table 6.3 gives the percentage true positive identifications analysed by scene, which are displayed graphically in figure 6.7 to 6.9.

While analysis of results by scene is undoubtedly an important part of evaluating the system variants, the process is complicated by the need to consider and decouple the instance specific

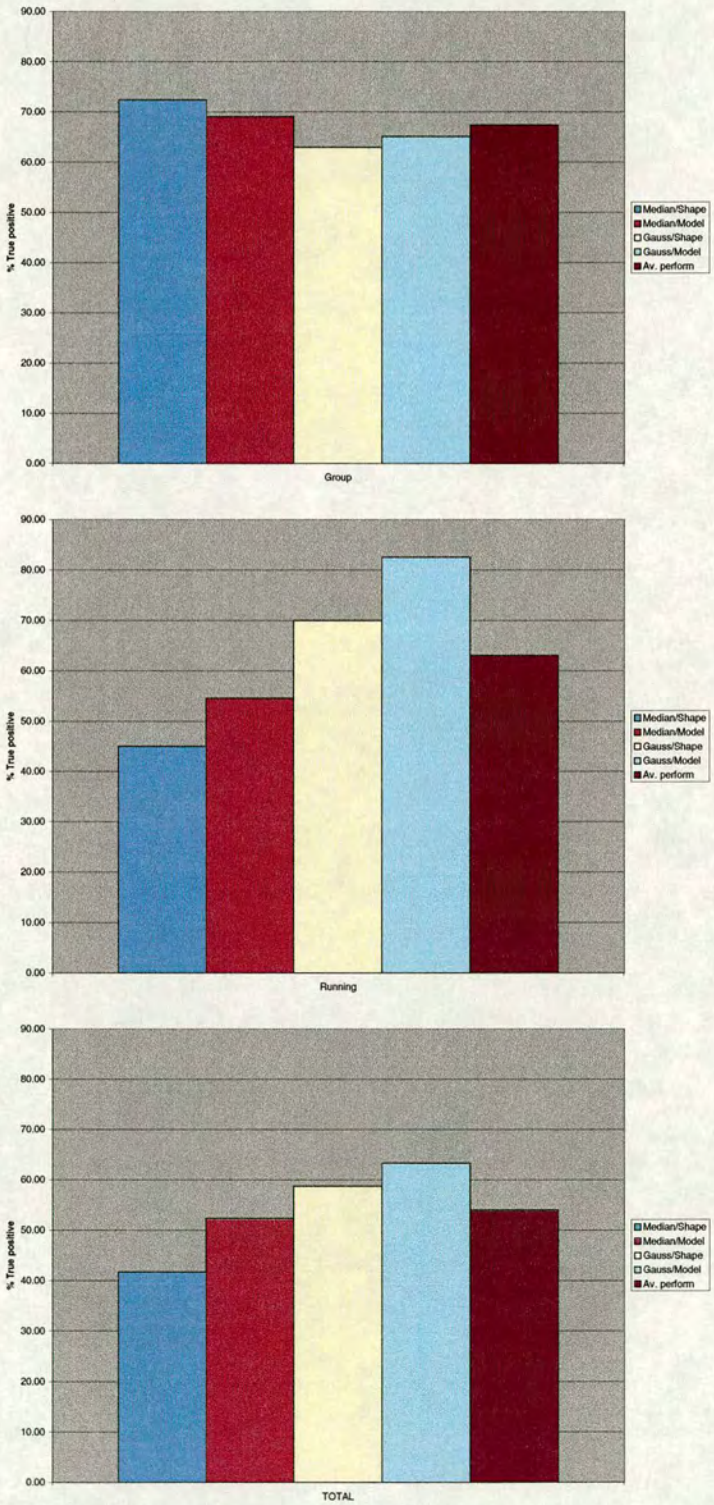

Figure 6.6: *True positive identifications for group and running instances and totals over all instance types.*

Scene	Median/ Shape	Median/ Model	Gaussian/ Shape	Gaussian/ Model	Average performance
1b	39.49	49.68	58.28	68.83	54.07
1c	36.52	45.30	46.67	38.26	41.69
1d	41.29	43.26	53.33	39.43	44.33
1e	26.83	47.56	64.63	74.49	53.38
1f	13.41	26.52	41.44	66.43	36.95
2a	49.39	55.49	54.32	58.54	54.43
2b	48.83	61.98	63.32	62.96	59.27
2c	56.38	64.45	62.69	68.82	63.09
3a	30.77	41.03	70.51	83.52	56.46

Table 6.3: *Percentage true positive identifications analysed by scene*

effects prevailing for each scene. It is nevertheless possible to make important and useful general observations concerning system variant performance in this context.

While it has been possible to give quantitative measures to assist the comparative analysis between scenes, the absolute values for any scene are also subject to the spread of instance types over the scenes. Thus it is feasible to make qualitative comparisons of patterns and overall performance after decoupling major instance biased effects. Without detailed analysis of each scene by individual instance however, it is not meaningful to attempt a full quantitative derivation of the exact values.

Scene 1b

It is very useful to consider this scene as a *control* against which to compare the performance of the system variants on other scenes. The lighting conditions are optimal in that there is full natural lighting with no gross illumination changes during the sequence. Visibility is good so there are none of the occlusive/noise effects associated with rain or snow. Wind is classified as mild, as opposed to strong, the latter characterised by the extent of attributable motion seen in vegetation and other flexible scene objects. Finally, scene activity is classified as sparse, in that the pedestrian events are generally discrete and do not involve group entities.

The pattern of system variant performances match the prediction based upon the totals over all instance types as seen in figure 6.7 in that is a regular increase of performance from the median/shape to the Gauss/model variants, with the greater improvement seen on enhancing the segmentation module.

Scene 1c

The clearest change for this scene is the overall drop in performance over all system variants, with an average rate of 41.69% (figure 6.7). The scene was recorded in failing lighting and the reduction in ambient illumination and the consequent reduction in contrast make foreground discrimination more difficult for both segmentation approaches. A possible solution to this problem is to make the thresholding for the pixelwise foreground/background decision a variable dependent on the average pixel variance for the image overall.

An anomaly can observed in the pattern of performance in the relatively low success rate of the Gauss/model variant. Further detailed analysis of the failure instances reveal a cause which constitutes a potential drawback to the mixture of Gaussians segmentation approach.

As previously discussed (section 4.4.4), segmentation using the match of each new pixel value with a corresponding mixture of Gaussian distributions allows greater flexibility in the background representation. Among other advantages, greater pixelwise discontinuity can be portrayed as illustrated in figures 6.3 and 6.4. In a low light scene, with more background variation caused by the local and directional nature of artificial lighting, this attribute is emphasised.

Divergence in the value neighbouring pixels develop as thresholds for assignment to background (due to slow divergence, not sudden enough to have caused the initialisation of a new background distribution) coupled with similar variation caused by increased shadowing can give objects with a most irregular outline pattern. This irregularity appears to be distributed quite evenly over the object boundary on average, which causes little problems for the shape based approach as the gross height/width ratio is generally unaffected. The model-based approach, however relys on a good percentage of matched boundary points around the object and so fares less well in these circumstances.

Scene 1d

This scene was recorded at the beginning of the day, with very similar conditions to those observed in scene **1d** at the day's close. We would thus expect to see results very similar to those of scene **1d** and this is indeed observed with an average rate of 44.33% (figure 6.7). There is no significance attributed to the variations in individual system variant performances, which do not exceed 7% and do not cause a deviation from the pattern observed for scene **1d**.

Scene 1e

This scene differs from the control in two main respects, a reduction in visibility caused by the

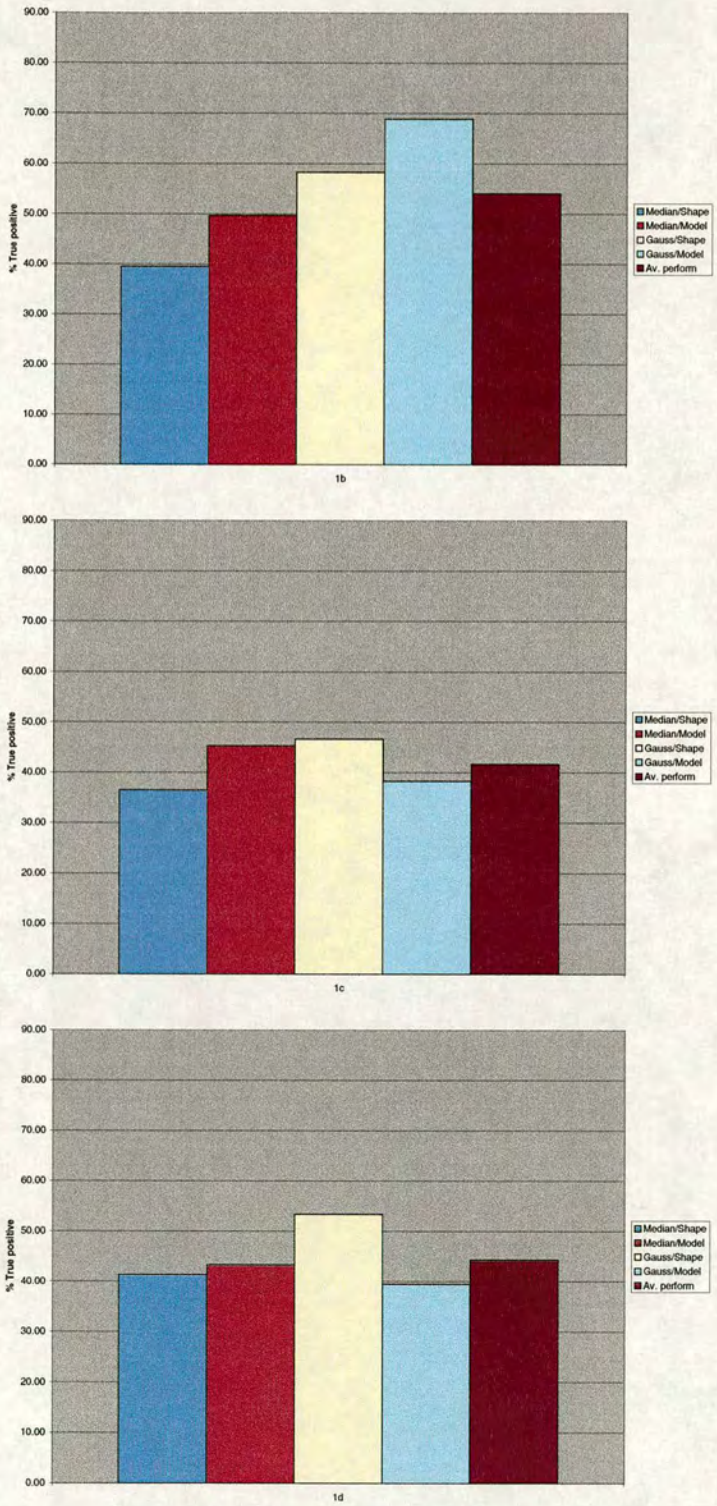

Figure 6.7: *True positive identifications for scenes 1b, 1c, 1d*

rain/mild snow mixture and a greater amount of pedestrian activity overall. It is interesting to note that the systems overall cope better with the occlusive scene effects than they did with the overall contrast reduction, the latter affecting the variants across the board at the most basic level of processing.

The prevailing meteorological conditions cause a mild occlusion to apply for all instances, even those nominally classed in table 6.1 as simple. This accounts for the greater variance between system variant performance with a low of 26.83% for the median/shape combination and a maximum of 74.49% for the Gaussian/model variant.

It will be seen (figure 6.8) that the absolute performance values of the Gaussian variants in this scene actually exceed those for the methods in scene **1b** (by an order of 5%) although that is classified as a less difficult scene. I attach no particular significance to this particular observation but it is mentioned to emphasise the point that random variances in absolute performance values of this order are to be expected given the low level of precision possible in the scene by scene analysis.

Scene 1f

This is a more extreme example of the complicating effects of meteorological occlusion across all incidences in a scene. With a greater rate of snow fall and larger constituent particles, the percentage occlusion superimposed on the constituent incidences is correspondingly greater. As expected, the result (figure 6.8) is a further reduction of overall performance (down to 36.95%) and a still greater variance in performance across system variants (now 13.41% for the median/shape combination and 66.43% for the Gaussian/model variant).

Scenes 2a, 2b, 2c

These scenes are considered together as the dominant effects in the results pattern apply across all three. The most noticeable feature of the results (figures 6.8 and 6.9) is the reduction in the improvement to results caused by switching from the median segmentation approach to the mixture of Gaussians version. The overall trend, in fact, is that of a levelling of performance abilities over the system variants.

The underlying cause of this shift is the same for all of the scenes: as can be seen in table 6.1 *group* incidences constitute a significant proportion of the makeup of each. Considering the performance pattern for group incidences generally (figure 6.6) what we observe is the superposition of this with the expected pattern of performance variance seen overall. We also

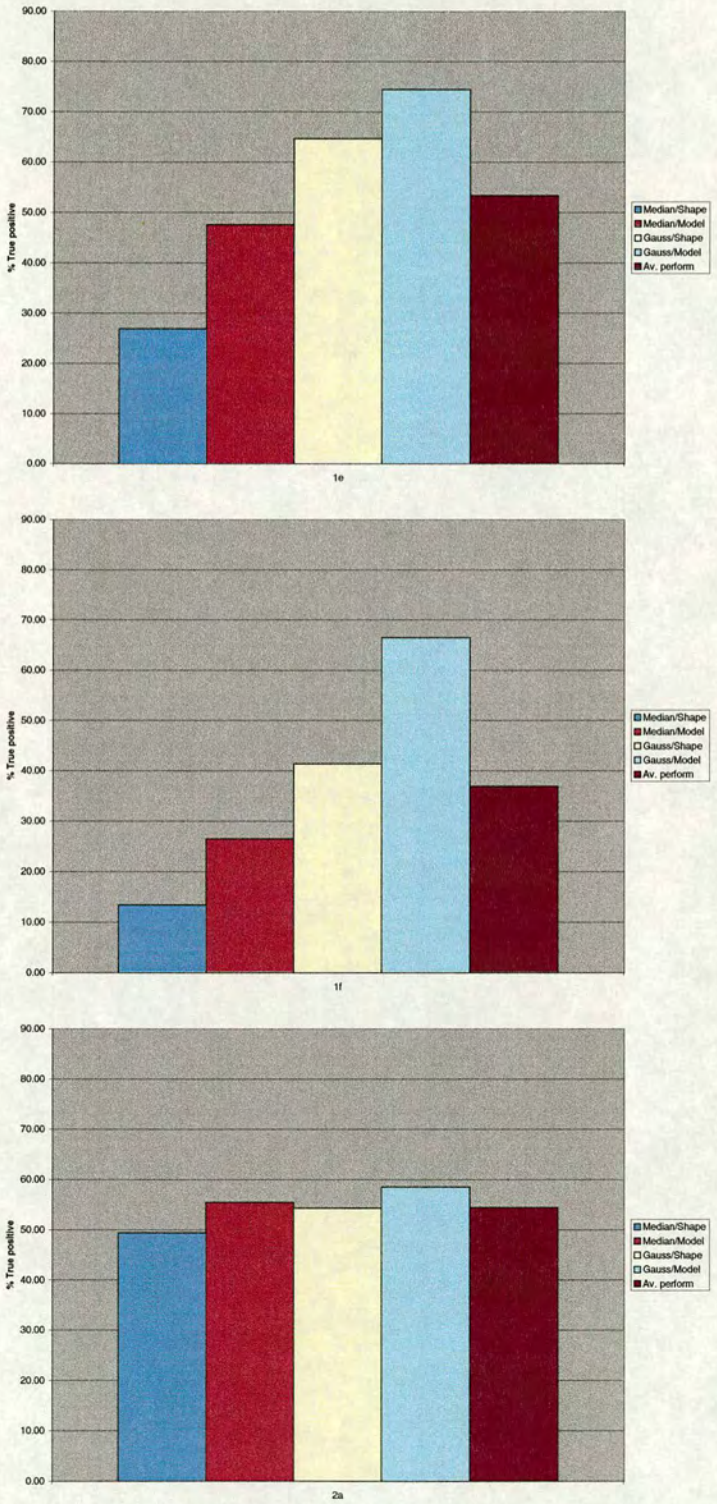

Figure 6.8: *True positive identifications for scenes 1e, 1f, 2a*

see that a greater proportion of group incidences reduces the average performance over all systems, which is reasonable given the change in the nature of the entity under observation.

What does **not** emerge from these results is a significant variation in results due to other changes in scene construction: the increase in wind speed from **2a** to **2b** and the increase in overall scene activity between **2b** and **2c**. The effects of the change in proportion of group activity effectively swamps any useful results on these scene variations. It would be useful to obtain further sequences which illustrate these variations in a scene without the high levels of group activity to analyse these effects better.

Scene 3a

The final scene is again similar to **1e**, with the variation being that there is only mild snow, without a rain component. The results are as anticipated (figure 6.9) with a very similar profile and superior overall performance.

6.3.4 Performance of system variants over all scenes

In summarising the relative abilities of the system variants, it is instructive to consider the profile of performance of each variant across all scene types (table 6.3, figures 6.10 and 6.11).

This firstly reinforces the fact that, as previously identified, performance results increase from median/shape to Gaussian/model. However where, in most cases, greater improvements are given by enhancing the segmentation module (especially in adverse meteorological conditions), the exception is as detailed for the low light scenes **1c** and **1d**.

There are broad patterns in the results (figures 6.10 and 6.11) : the worst performance is *generally* for scene **1f**; that for scene **3a** is always superior; performance on scenes **2a**, **2b**, **2c** is similar. However it is more significant to note that there is **not** an overall general pattern which can be identified for each variant across the scenes. It is **not** possible to state without qualification that the mixture of Gaussians segmentation approach is superior to the median nor that the model-based classification approach is superior to the shape-based.

The relative performances in terms of true positive identifications can only be stated for a specified instance type and even then scene conditions can give unpredictable variances. In evaluating performance for a robust surveillance application, it is necessary to both make assumptions about the scene and incidence variation and to consider the uncertainty inherent in

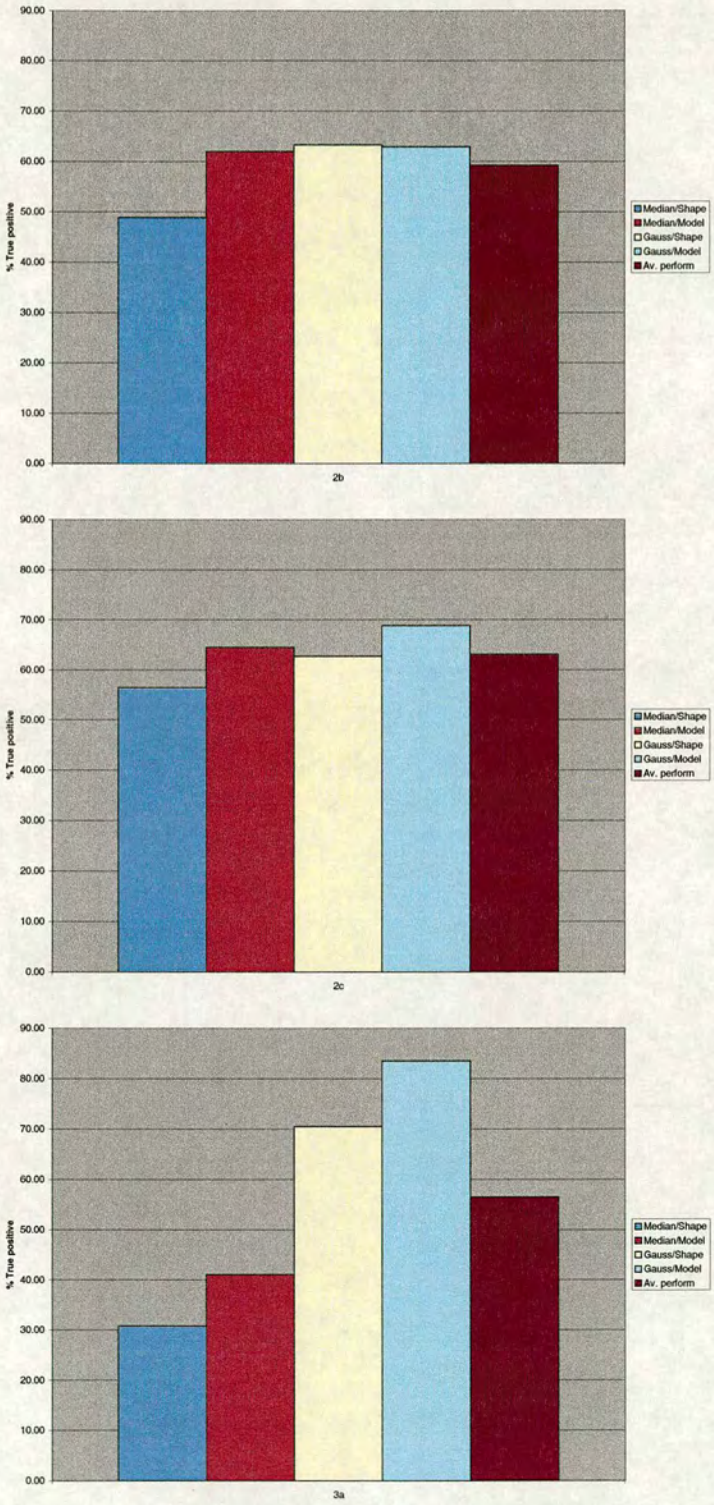

Figure 6.9: *True positive identifications for scenes 2b, 2c, 3a*

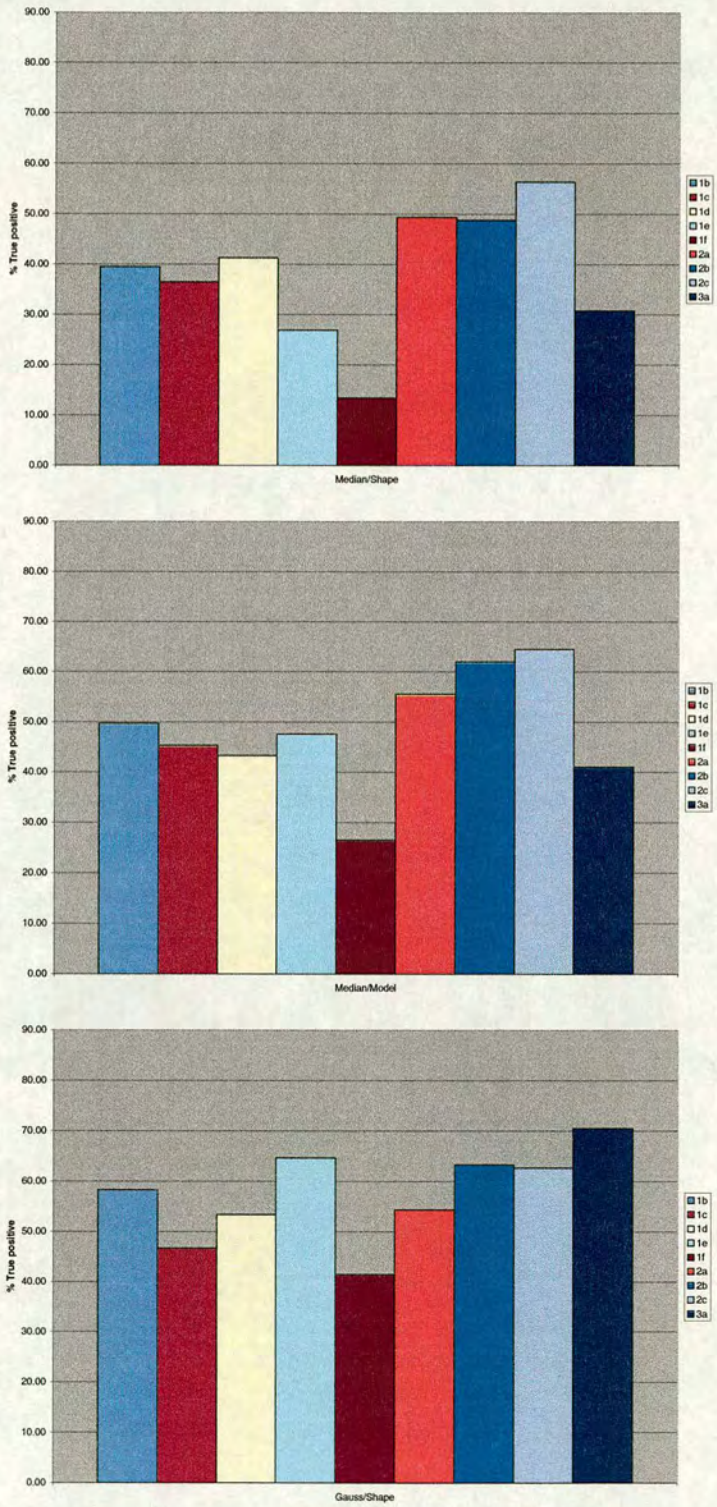

Figure 6.10: *True positive identifications for all scenes for median/shape-based, median/model-based matching and Gaussian/shape-based module combinations.*

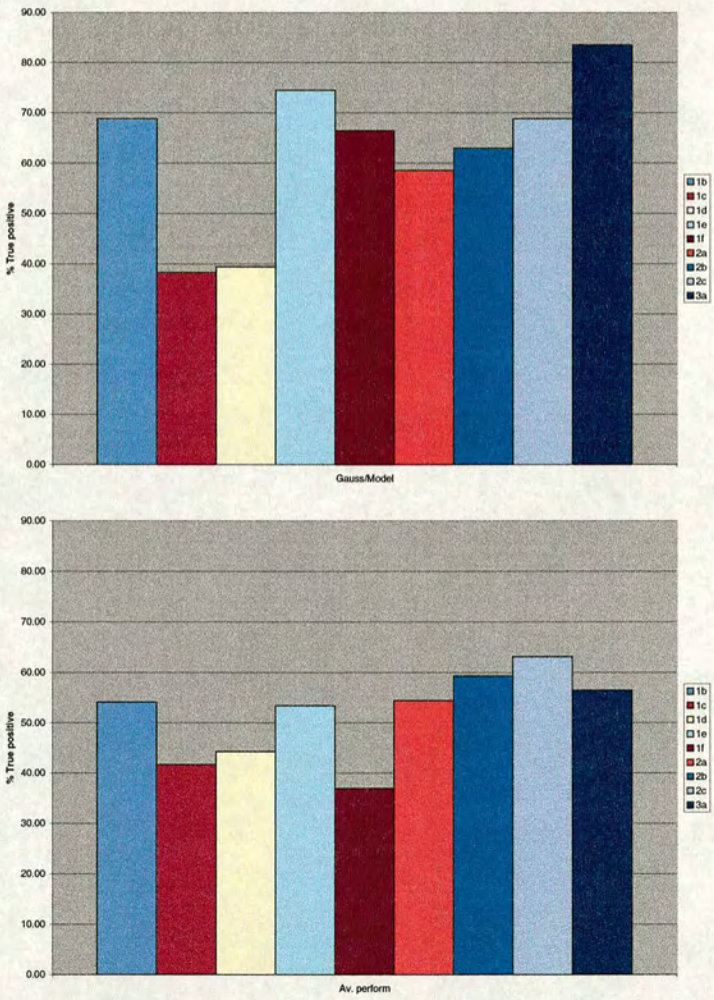

Figure 6.11: *True positive identifications for all scenes for Gaussian/model-based module combination and average over all methods*

Instance type	Median/ Shape	Median/ Model	Gaussian/ Shape	Gaussian/ Model	Average performance
Simple	0.33	0.24	0.54	1.96	0.77
Occluded	0.61	0.91	8.24	5.11	3.72
TOTAL	0.93	1.15	8.78	7.06	4.48

Table 6.4: *Percentage false positive identifications analysed by instance type*

the performance predictions made.

6.3.5 Analysis of true positive against false positive recognition percentages

As discussed in Chapter 5, it is important to consider a second way in which the system variants can fail to give the correct results. Analysis of scenes for surveillance purposes can be considered to be an instance of *binary hypothesis testing*: for any frame the *null hypothesis* can be that there are only background pixels/objects present, the *alternative hypothesis* that there is a pedestrian object in the scene. In binary hypothesis testing two kinds or errors can occur: accepting the alternative hypothesis, when the null hypothesis is correct and accepting the null hypothesis when the alternative hypothesis is true. The first error is often called *Type I* error or *false positive* and the second error is usually called *Type II* error or *false negative*.

Receiver Operating Characteristic (ROC) graphs are very useful in assessing the overall behaviour and reliability of the system in terms of these two error types. The ROC graph as adapted for the current system shows the relation between the true positive percentage (100 - false negative percentage) on the x-axis and the false positive percentage on the y-axis. The point $(100, 0)$ is the perfect classifier: it classifies all positive cases and negative cases correctly. The point $(0, 0)$ represents a classifier that predicts all cases to be negative, while the point $(100, 100)$ corresponds to a classifier that predicts every case to be positive. Point $(0, 100)$ is the classifier that is incorrect for all classifications.

An ROC curve is often plotted on such a graph as tuning parameters are varied for a single system, to ascertain its effect on the two types of error. We modify this approach to present data on true/false object recognition for each of the four system variants as a point in a subsection of the ROC graph. The ability of each variant is proportional to the distance from point $(0, 100)$, with different weightings attached to the X and Y components dependent on the importance attached to the two type of errors.

Scene	Median/ Shape	Median/ Model	Gaussian/ Shape	Gaussian/ Model	Average performance
1b	2.42	0.00	5.63	3.05	2.77
1c	0.00	0.82	5.05	5.56	2.86
1d	0.00	0.56	0.00	8.85	2.35
1e	1.20	1.99	20.13	12.90	9.06
1f	1.59	1.63	28.74	23.53	13.87
2a	2.96	4.09	4.71	7.87	4.91
2b	0.00	0.00	0.47	1.35	0.45
2c	1.05	0.43	4.03	6.46	2.99
3a	0.00	0.00	1.27	1.09	0.59

Table 6.5: *Percentage false positive identifications analysed by scene*

True/false positive identifications analysed by instance type

Table 6.4 gives the false positive identifications analysed by instance type. Each diagram in figure 6.12 shows a plot of the ROC graph area between 0% and 90% for the true positives and 0% and 10% for the false positives. The results are shown separately for *simple* and *occluded* (there being no false instances in the data for the other types) as well as over all instance types.

As expected, the results are better overall for simple instances but the general pattern is common across the instance types: the optimum method will be either median/model (true $= 52.35\%$; false $= 1.15\%$) or Gaussian/model (true $= 63.31\%$; false $= 7.06\%$) dependent on the weighting assigned to the error types.

True/false positive identifications analysed by scene

To get an idea of the spread of behaviours for the system variants in terms of the false/true positive results, it is useful to consider the performance scene by scene. Table 6.5 gives the false positive identifications analysed by by scene.

Figure 6.13 presents a graphical summary of these behaviours, where the key factor to be illustrated is not the individual values, but how the spread of the results for the methods varies from scene to scene. An unlabelled plot of the points for all scenes together is given to illustrate the overall spread of results.

This clearly illustrates a further drawback of the mixture of Gaussians segmentation approach: the false negative rate is consistently higher than that given by the median segmentation method.

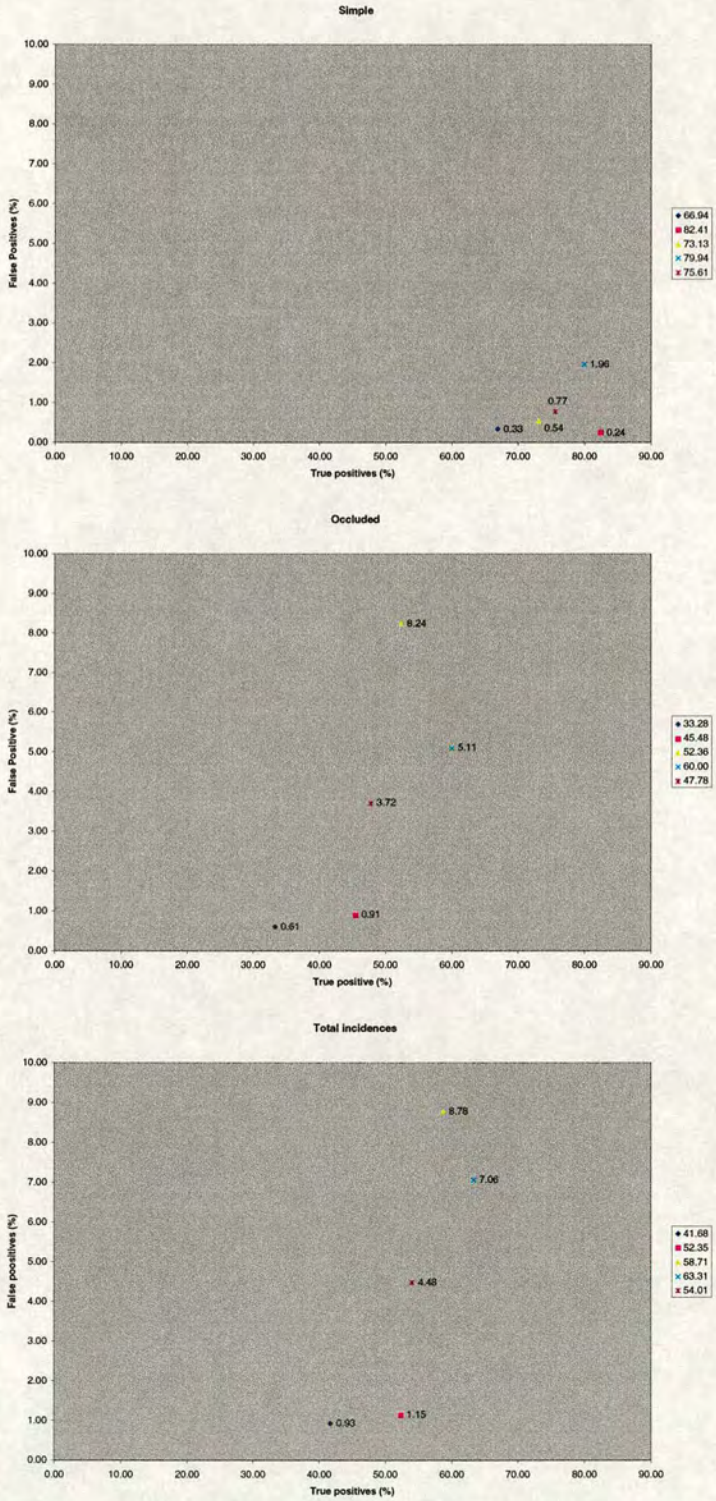

Figure 6.12: *ROC plots for simple and occluded instances and totals over all instance types.*

The effect is most pronounced in scenes with poor visibility due to meteorological conditions (**1e** and **1f**) where aggregations of random scene variation caused by the increased scene noise can be incorrectly interpreted as significant moving objects.

Performance of system variants over all scenes

In summarising the relative abilities of the system variants using this metric, it is again instructive to consider the profile of performance for each variant across all scene types (figures 6.14 and 6.15).

As for the similar analysis for true positives alone, the difference in performance results is most notable between the two segmentation approaches. However, in this analysis of results it can be seen that it is misleading to characterise performance change as a linear increase in performance from the median to the mixture of Gaussians approach. While it remains evident that the points do indeed cluster around a higher true positive value in the graphs both Gaussian segmentation based approaches, this point also has a significantly higher false positive value.

In evaluating these results it would again be necessary to draw on the application specific information as to the relative significance of the values for true and false positives. In the application for which the current evaluation is being made, there are several issues to consider.

The relative importance of a good true positive rate is dependent to some degree upon the specific surveillance application to which the system is applied. It is important to be able to maintain a consistent track of a pedestrian if useful analysis is to be made of their trajectory but to obtain simple alarm data based on their hypothesised presence in a specific area, obtaining just one true positive incidence is sufficient. As noted in section 6.3.2, the current performance of the system overall in respect of true positive rates is not optimised and a conclusive decision in this respect is thus harder to form.

The situation is clearer in respect of false positive rates: one of the key attributes required of the system is that it run efficiently within the context of a distributed application whose operation is characterised by the selective transmission of alarm data over a network. The system has been biased towards minimising false positives to reduce unnecessary transmissions which both tie up network band width and reduce response efficiency in true alarm situations. There is thus a heavy weighting in favour of reducing the false positive rate to a minimum while maintaining adequate true positive response.

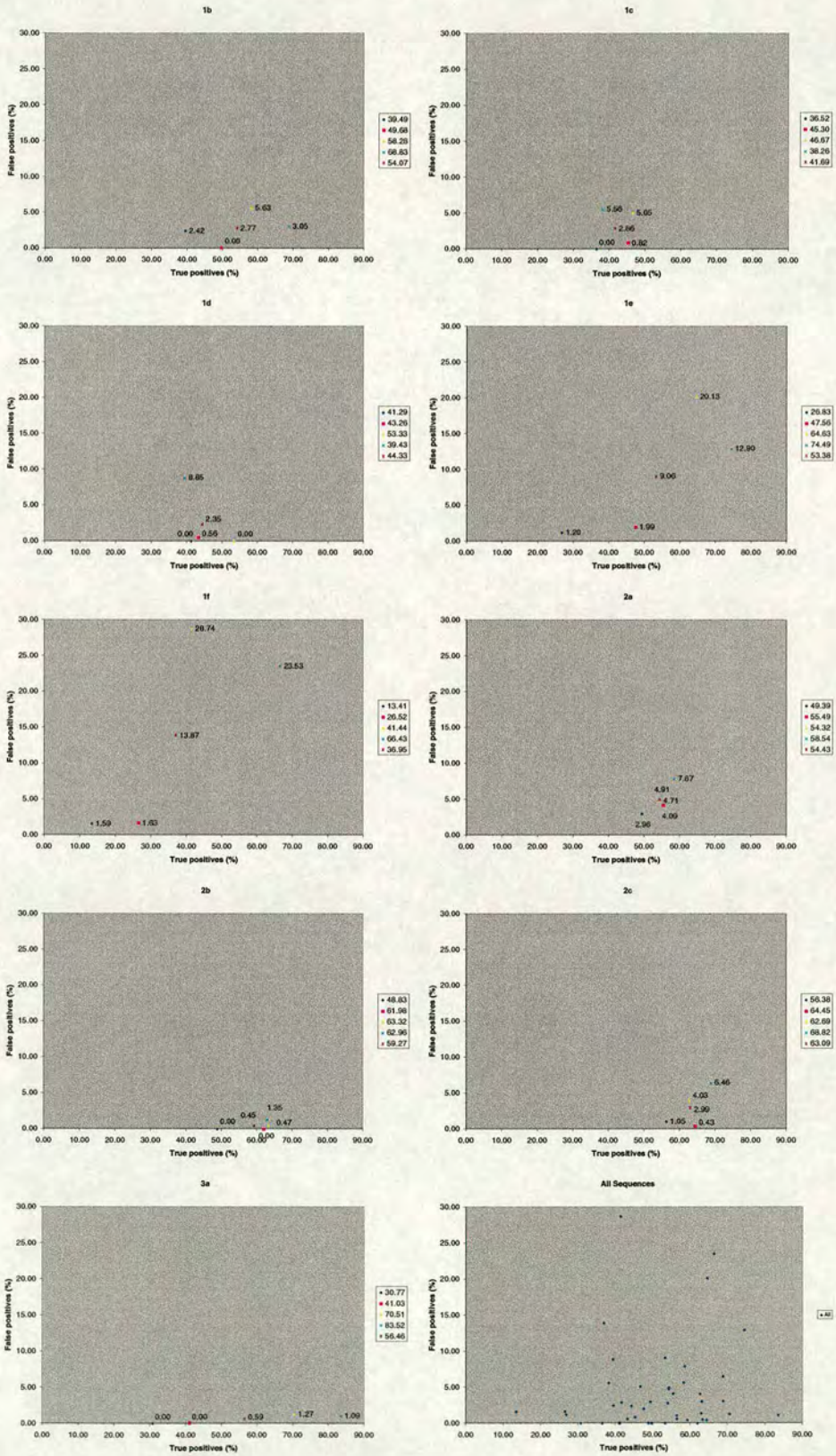

Figure 6.13: *ROC plots each scene individually and summation for all scenes*

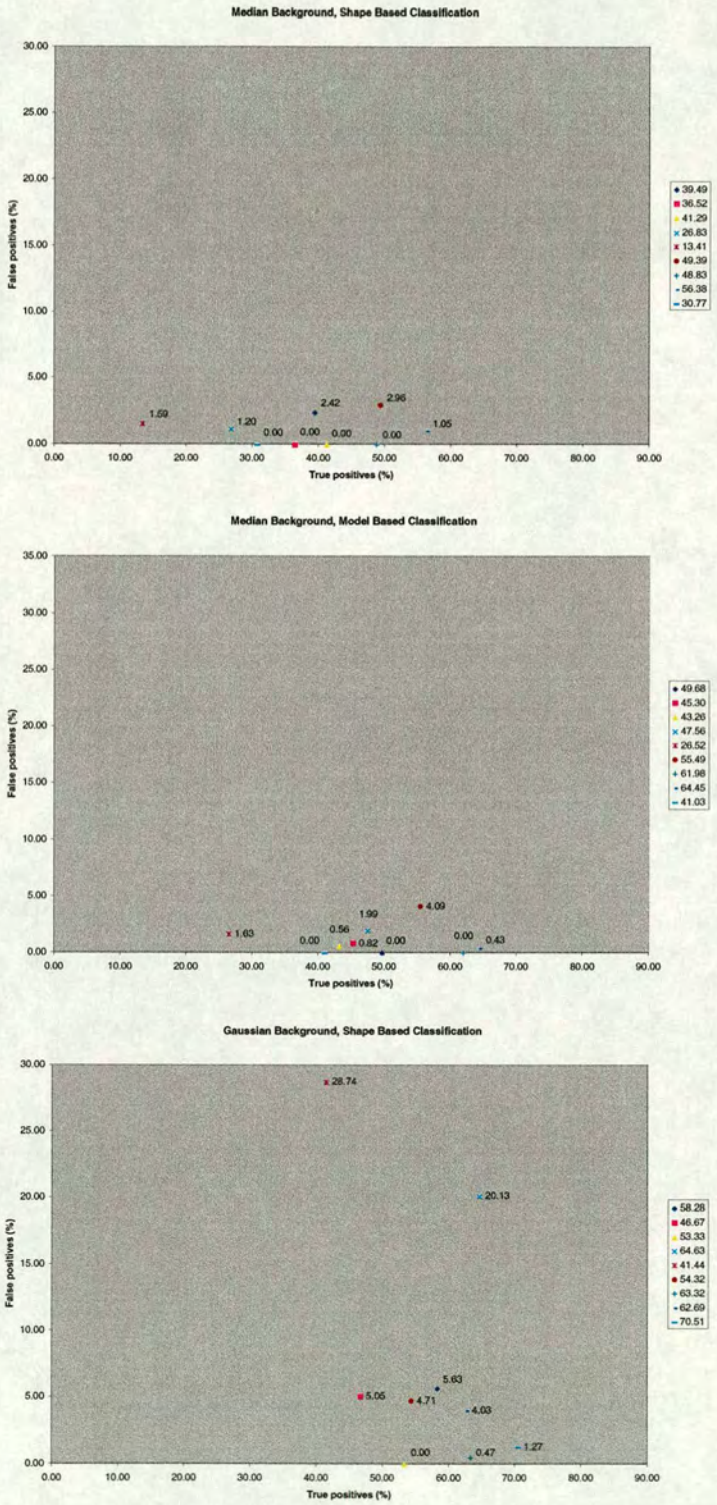

Figure 6.14: *ROC plots for median/shape-based, median/model-based matching and Gaussian/shape-based module combinations.*

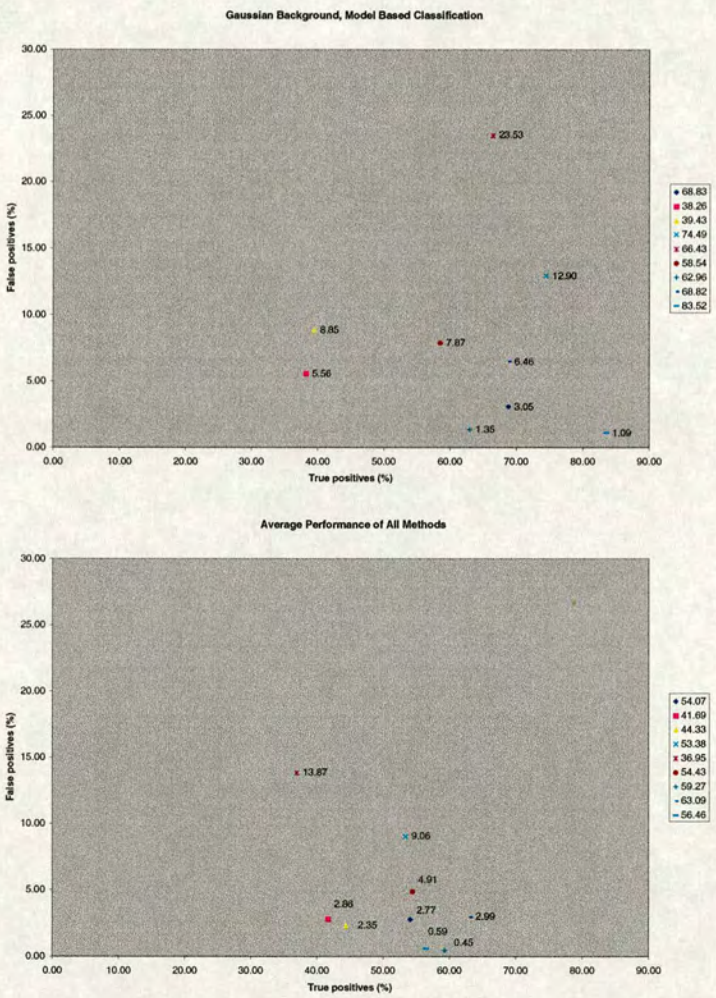

Figure 6.15: *ROC plots for Gaussian/model-based module combination and average over all methods*

Module	Average(s/f)	S.D. (s/f)
Initialisation	36	9
Median Segmentation	4	2
Gaussian Segmentation	11	7
CCA object growing	96	80
Edge trace object growing	0.7	0.5
Shape-based classification	0.2	0.1
Model-based classification	0.5	0.3

Table 6.6: *Module timing results in seconds/frame*

Considering the results in this context, the relatively high maximum of false positive rates seen over the mixture of Gaussians variants (28.74%) are unlikely to be acceptable as compared to that for the median approaches (4.09%). As noted above, these excessive rates occur for the scenes with poor meteorological conditions (**1e** and **1f**), but even if these were excluded, the high maximum seen over the mixture of Gaussians variants is still a factor of two higher at (9.06%). Based on these results, without considering possible modifications possible by using supplementary processing modules, the Gaussian background approach appears less suitable for the intended application.

6.4 System resource requirements

The balancing factor to be considered against the level of performance associated with a system variant is the resource requirement of that variant. The most effective combination of modules would be of little use if they are too memory intensive to run in the operating platform's resource scarce environment or if they run so slowly that results cannot be transmitted on an approximately real time basis. Each system variant was therefore timed during its operation and the maximum memory required registered as detailed in Chapter 5.

6.4.1 Comparative analysis of system variant run speeds

The number of system variants to be considered in this context is eight, as each of the four variants analysed for performance characterisation can run with one of two *object growing* methods. As the two object growing methods give the same intermediate output they do not have an impact on the performance results and so were not considered in section 6.3.

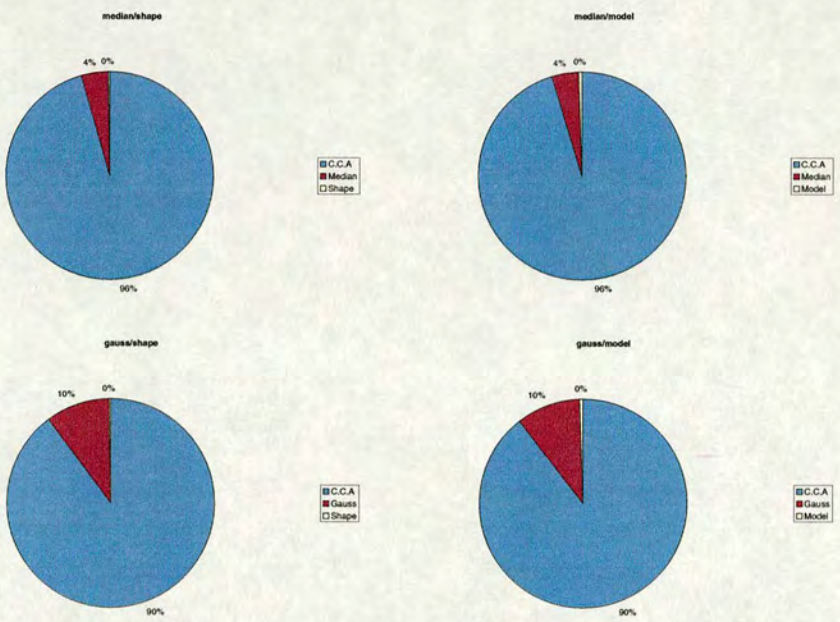

Figure 6.16: *Charts of relative time cost of modules using connected component object growing*

The systems were first developed using a *connected component algorithm (CCA)* based object growing method (section 4.5), using which the bulk of the testing was performed. It became clear that this step was making a disproportionate draw on processing requirements and so an alternative *boundary tracing* object growing method was developed for repeat runs.

The timing results for the individual modules is given in Table 6.6, specifying the average and standard deviation of the figures. The data spans the constituent frames of the sequences for each frame, with a relatively large standard deviation due to the effects of the large variation in scene complexity over the data sample.

The first clear result is that system variants employing the connected component algorithm based object growing method are unlikely to be suitable for use in a real time surveillance application. With an average per frame processing speed on a dedicated workstation of 96 seconds per frame, the variants cannot output data at a serviceable rate for such a use.

Considering the charts (figure 6.16) displaying the processing time spread over the component modules (excluding initialisation) for the variants, it is clear that varying the other modules will have an insignificant effect on processing speeds, as they account for only 10% of the time

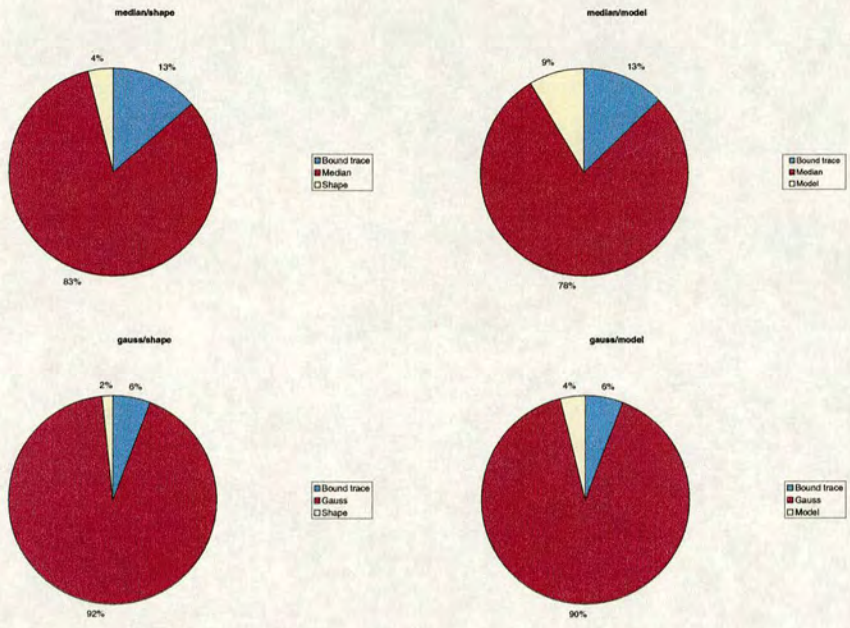

Figure 6.17: *Charts of relative time cost of modules using outline tracing object growing*

taken.

Focusing attention on the approaches employing the boundary tracing based object growing method then (figure 6.17), the most significant choice in terms of time resource utilisation is of the segmentation module. Accounting for up to 90% of the processing time of the system variants, changes in the time taken by this will be the critical factor in controlling operating times.

It can be seen that the mixture of Gaussians segmentation approach takes on average seven seconds per frame longer to give a result than the median based method. This can represent more than a doubling of the overall processing time per frame, which must be a significant cautionary element in considering the merits of this approach.

The shape-based matching approach is on average twice as fast as the model based approach, but as the matching step only accounts for from 2% to 9% of the time per frame, this turns out not to be so significant a consideration in evaluating the optimum overall variant.

System variant	heap (kb)	RSS (kb)
Median/Shape	153.2	153.2
Median/Model	153.8	154.0
Gauss/Shape	174.3	178.3
Gauss/Model	175.3	179.6

Table 6.7: *System variant memory requirements*

6.4.2 Comparative analysis of system memory requirements

The memory requirements of the system variants (table 6.7) represent the maximum amount of memory recorded as used by a system variant during the runs made over all scenes. Again, no variation was observed by varying object growing methods and so this variable is excluded from consideration.

The key variable is, once again the choice of segmentation methodology, with the object classification approach making little alteration to the overall resource demands. Looking at the memory used in the heap, the variation from 154kb to 175kb is only 14%, but in the practical application where memory is at a premium, this can nevertheless be a significant factor.

6.4.3 Mapping of resource requirements to operating platform

Although the performance results are useful not only as comparative measures, but also as valid projections of ability on the final platform, the same does not apply to resource requirement measurements. As they stand, these are useful in their primary purpose only, that of concluding the most suitable module combination to be transferred to the final platform.

Both timing and memory usage statistics are inherently platform dependent and no universal approach exists to reliably map the statistics between arbitrary platforms. Using the *operations per second* performance of two systems to project times is only suitable to give an approximate mapping as it cannot encompass deviations due to variation in efficiency of different CPU architectures to perform diverse tasks. Also, factors such as memory access times (whose significance changes in relation to the number of transfers required, the type of memory used etc.) vary widely between systems

Even the basic memory usage data on one system can be very difficult to interpret meaningfully where, as in the example under consideration, one system uses virtual memory and states

memory usage in terms of resident set size which has no direct correspondence with memory usage on a simpler system.

SPEC (Standard Performance Evaluation Corporation) have the stated mission:

> *To establish, maintain, and endorse a standardised set of relevant benchmarks and metrics for performance evaluation of modern computer systems*

and make available benchmarking results for many common platforms, running a variety of applications [116]. Their use is clearly limited in respect of mapping resource usage to comparisons between instances of these common platforms and applications, and so are of little use in evaluating the performance of a novel application on a non-standard platform.

The original intention was to physically transfer the optimum system variant onto the final platform as the final stage in the performance characterisation, there to analyse its speed of operation and memory usage in situ. This extension of testing had to be abandoned due to a combination of an error in communication at the system specification stage and attendant time constraints.

The final system uses a unique and proprietary operating system, *CamOS* which uses what is is described as a *pseudo-C++* compiler. This fit in well with the intention to develop the system in C++ taking advantage of its object oriented nature to facilitate modular design. The code was to be developed in standard C++ using the g++ compiler and gnu programming tools and would then be modified in respect of the unique features required under CamOS.

It was not until the bulk of the system had been implemented that it became clear that the CamOS compiler was not in fact configured to support C++ style coding conventions. Transfer to the final system will involve substantial rewriting of the system code using C programming conventions before being further modified in respect of the unique features required under CamOS.

At this stage it was understood that this would require a significant input of man hours and it was considered no longer feasible for inclusion within the scope of the current work, necessitating recourse to the previously noted approximate mapping approaches.

Only during the final stages of the project, at the very end of the write-up process, was it realised that most C++ compilers can generate 'vanilla' c as a first pass preprocessing stage. This could

Module	Average(s/f)	S.D. (s/f)
Initialisation	363	91
Median Segmentation	40	20
Gaussian Segmentation	111	71
CCA object growing	967	807
Edge trace object growing	7	5
Shape-based classification	2	1
Model-based classification	5	3

Table 6.8: *Projected timing results on final platform in seconds per frame*

be precessed by the CamOS compiler with relatively small amounts of modification. At this point it was not feasible to attempt this additional process, but the lessons learned from this compounded error are discussed in section 7.3.1.

Despite the limitations of the approach, as noted above, it is still possible to give an rough mapping for the operation of system variants on the final platform from that on the development platform by neglecting the complicating factors to give a strictly approximate measure.

From Chapter 3, the VideoBridge VP400 In-camera codec uses is a 33MHz 32bit RISC processor with 1Mbx32-bit SRAM memory available for storage of code, data and the heap. The development platform, the Sun Microsystems Ultra 10, uses the UltraSPARC-IIi processor, which has a processing speed of 333MHz. It utilises virtual memory and the units used in testing were part of the Department of Electronics and Electrical Engineering network.

As noted in Chapter 5, testing was conducted overnight during a holiday period: the department system default settings default remote users to unused workstations and so the probability of significant distortion of timing data by other usage was low. Workstation job statistics was monitored on an approximately hourly basis as a supplementary check on usage status. The test data was gathered over 32 runs for each module further reducing the effect of possible distortion of data by any brief or intermittent additional load on the workstations.

In the circumstances, it is only reasonable to make the most approximate computation of projected processing times by applying the ratio of the two systems' processor speeds to the test platform results. Similarly, the only basic hypotheses available on memory usage is to assume no difference between the dynamic memory requirements in the two systems. The very approximate results are presented in tables 6.8 and 6.9.

System variant	Heap (kb)	Remaining DRAM (kb)
Median/Shape	153	359
Median/Model	154	358
Gauss/Shape	174	338
Gauss/Model	175	337

Table 6.9: *Projected memory usage on final platform*

Dependent on the accuracy of these projections it appears that the system requires further optimisation if it is to be of practical application on the final platform. Several possibilities exist to reduce resource requirements which could apply to any system variant.

The background update need not be performed on every frame registered by the camera: the system could be set to perform background update only every n frames, reducing the average processing load per frame. This would clearly have an impact on the performance of the system to the degree that the background would adapt more slowly to environmental changes.

The architecture of the system allows parallel processing of multiple tasks with message passing to transmit results between these tasks.It should be noted, however, that the RISC itself is restricted to multitasking, so parallel processing proper is not available for routines which run only on the RISC. Rather than specifying background update only every n frames, the background update process could run in parallel with the object tracking. The background would be updated ready for the next captured frame which would would then be used for the next background update. Again, this option reduces the average processing load per frame but the background update process operates continuously at the maximum speed which processing resources allow. A disadvantage is the processing overheads for the message passing and other parallelisation factors.

6.5 System extension results

As noted in section 4.8.1, a proposed extension to add value to the system on the final platform is to use a camera's control input to automatically direct pan, tilt and zoom to capture a high definition face image. The camera would be directed to that portion of the scene corresponding to the position of a face according to the object classification scheme, using a simple geometrical assumption as to the average human form.

Figure 6.18: *Face images extracted from a sample sequence containing one pedestrian and an excerpt image from the sequence*

Although not implemented in full, a pilot routine was constructed to illustrate the feasibility of the approach, using the simple shape-based object classification approach. Rather than direct a PTZ camera, the hypothesised face area was specified using this model in terms of a sub window in a short sample sequence of images containing a single pedestrian. The results of this simulation are given in figure 6.18. The face images are of a lower resolution than would be the case if the approach were applied at capture, but the approach can be seen to be suitable for directing attention to the appropriate area in an image in this case. A suitable face image is captured for most frames, but the last two extractions illustrate a limitation of the approach where the face is lost. This could be overcome simply by storing a subset of the face captures equivalent to that displayed for this sequence, increasing the probability that a useful face shot was obtained.

For a real-time image capture version, some correction would need to be made for predicted motion in the body during the re-direction time of the camera. The criterion for selecting the individual whose face is to be captured could be provided by some heuristic rule set related to the alarm condition or via a user interface.

6.6 Summary

The performance characterisation on the development platform has given some unexpected and interesting results. The qualitative predictions were that results overall would be superior performance from the Gaussian/model variant over the median/shape version, with intermediate results from the other two variants. What has emerged is that although this overall pattern indeed holds in many situations it is not a clear cut relationship and is subject to provisos and limitations.

Considering the overall results presented in tables 6.2 and 6.3, the first evident point is that the overall positive identification rate does not exceed 75% for any scene. Further, the positive identification rate even for simple instances does not exceed 83%. Compared to results presented in the literature and to the level of performance required from a system with practical usefulness, this level of performance is poor.

In respect of the discrepancy with results in the literature, it is worth noting that the present system is being tested thoroughly over a large data set including diverse scene types. The tuning of parameters which can occur in systems tested on small data sets and which may contribute to the published results are thus eliminated, making direct comparison difficult.

In terms of the performance desired in a commercial system not being achieved, it should be noted that the implementations under examination have not been optimised. All modules were tested for correct functionality and then immediately brought into the system for comparative testing. Only the system variant suggested as appropriate for the final platform would be optimised in parallel with its transfer onto that platform. The performance improvement cannot be quantified in the absence of this stage's completion.

From section 6.3.2, the best overall performance in terms of true positives over all instance and scene types is indeed from the Gaussian/model variant. The biggest enhancement in system performance comes from changing the segmentation module to use the mixture of Gaussians

approach.

Segmentation module

Considering the *mixture of Gaussians* segmentation approach in detail, it clearly performs best in instances of median occlusion (and thus also in rain/snow) and results from previous work [48] suggest superior performance in scenes with a large degree of oscillatory motion generally. Furthermore, the latency in adjusting to new objects in the background is lower by at least a factor of four (section 6.2).

This segmentation approach performs quite poorly however in 'twilight' scenes with artificial lighting and low levels of background/foreground contrast (section 6.3.3). In surveillance applications, these 'twilight' scenes can be of particular importance as they often correspond to times of high risk for intrusion. Miscreants will tend to take advantage of the camouflage opportunities inherent in poor light conditions. A system which is not robust to these conditions would thus be at a great disadvantage.

A further drawback for this segmentation approach is the relatively high false positive detection rate observed across all scene types (table 6.5 and figures 6.14 and 6.15). As an explicit goal for this system is to minimise false alarm signals, this too is a significant failing.

Finally, use of this segmentation approach increases time resource demands by the order of 100% (table 6.6) which, using current approximations (table 6.8), equates to more than a minute per frame on the final platform.

Given this extensive list of drawbacks to balance against a projected 17.03% increase in true positive detections overall (section 6.3.2), the mixture of Gaussians segmentation approach would not be selected in favour of the median average version.

Classification module

Considering the shape-based classification approach in terms of resource requirements first, the model-based approach operates on average half as fast as the shape-based version (table 6.6) which, assuming the median segmentation approach is used, equates with a 6% increase in processing time per frame. In terms of overall performance, the model-based version gives a 10.67% improvement in true positive identifications (section 6.3.2) and a maximum false positive rate only 1.13% higher.

While the shape-based classifier has proven to be better in coping with bisective occlusion the

possibility of double counting in these instances mitigates the benefit somewhat (section 6.3.2). Further, the median/shape variant system encounters significant difficulties in scenes with a large amount of ground shadowing (section 6.3.2).

Suggested system variant

It is necessary to take into account all of the above specific considerations in concluding on the optimum choice for development on the final platform. This vital specific data was derived from a detailed quantitative analysis of the performance and resource requirements of the system variants over multiple instance and scene types.

Taking all these factors into consideration it is possible to conclude that the optimum is **not** the variant suggested by initial qualitative characteristics (the Gauss/model variant) *nor* the fully minimised system using the most economical approach to processing (the median shape/variant) but is in fact the median/model combination, using the boundary tracing object growing module.

This system does not give the best performance across all instance types, but the more robust all-round performance over scene types, reduction in false positive signals and significant reduction in time resource requirements serve to outweigh this fact for the specific application in consideration.

Chapter 7
Summary, further work and conclusions

7.1 Introduction

This thesis proposed that an integrated design and characterisation methodology for the construction of vision systems for a specified application, based on object oriented design principles, can efficiently provide a recommendation as to an appropriate implementation and that such a methodology can also contribute to an evolving database of test sequences and comparative performance information of more general use to third parties.

To establish these contentions, several goals had to be met.

A survey of existing work in the fields of vision system performance characterisation and those areas of vision research relevant to the application designed is presented in Chapter 2.

A methodology for the integrated design and characterisation of an appropriate practical solution to a specified vision processing problem was to be created and evaluated. The vision processing problem and design principles are introduced in Chapter 3 and the characterisation phase was detailed in Chapter 5. An evaluation of this methodology is presented in section 7.2.

The methodology was to be used to design a modular surveillance application to run on a distributed processing architecture, using object oriented design principles to implement a selection of standard algorithms and novel approaches. This process is detailed in Chapter 4.

The *system variants* of the modular surveillance application so designed was to have its performance characterised on real surveillance image sequences. The results of this characterisation are given in Chapter 6 and are discussed in section 7.3.

A relative analysis of the performance of the *candidate solutions* of the performances against manually acquired ground truth and review of resource requirements was to be performed. The results from Chapter 6 are considered in these terms in section 7.4.

The data base of real surveillance test sequences made available to third parties is described in Chapter5.

There are several conclusions to draw from this work. First, the design and evaluation methodology should be assessed to ascertain the quality of results and their generality compared with the resources required. Following on from this, the overall results for the test design problem itself can be evaluated in terms of their reliability and generality. Finally, the performance results for individual modules within the context of the system allow some conclusions to be drawn as to their relative effectiveness.

7.2 The design and characterisation methodology

The proposed integrated design and characterisation methodology is intended to bridge the gap between monolithic vision system design and testing and the anticipated use of detailed algorithmic performance characterisation data-sheets. By implementing a subset of individual modules for specified vision processes and evaluating the performance of composite system variants on real data, a choice of the most suitable combination for the current problem can be made.

The overall idea of using the framework of a cost versus benefit analysis for evaluating vision systems is complicated by both platform and application dependencies. General cross-platform results in respect of resource utilisation require an extensive characterisation regime to give quantitative data. An approach which is constrained to use less detailed investigation will only give qualitative results cross-platform, a significant limitation.

In respect of application dependency, the modular construction in terms of commonly used processes will allow some useful extrapolation of results to similar design problems. This extensibility would increase if a library of such evaluations evolved incrementally.

7.2.1 Individual module implementation

The individual module implementations are tested in isolation using a limited data set and qualitative evaluation of the results by a human observer. Any tuning parameters in the individual module are varied to give the optimum results in these terms.

This represents a weakness in the approach as one of the key assumptions underlying the need for this sort of overall evaluation is that human interpretation is not adequate for assessing such intermediate data. An anthropocentric bias is introduced that has no basis where the 'user' will be another automated process. Furthermore, there is no evaluation of the stepwise error levels which may be significant in overall system performance.

Some of these limitations represent the cost of the approximation required to reduce the resource requirements for the methodology. The approach is intended as a compromise between costs and benefits and so some sacrifice in results quality is to be expected.

One alternative approach to simply fixing the implementation at this stage would be to record relevant metrics (possibly in the form of ROC curves) of each module's performance in terms of the internal parameter settings. This could increase the generality of the eventual results, giving some indication of how to modify the system for different problem requirements.

7.2.2 Procedural abstraction

The initial intention was to enforce procedural abstraction for the individual modules, so that it would be possible to exchange/replace individual solutions independently. A strict application of this rule does not appear to be feasible for the current problem. To take an example, where the boundary-following approach was used to evaluate object size in the model-building step of a system, the same approach must be used in the labelling of the current image for correct operation.

This does not invalidate the approach, but it does make it more complex to operate, with the requirement for flagging as to which approach has been utilised. The use of design principles based on Object Oriented principles is still important and should be supplemented with sufficient documentation to specify where flagging is required.

7.2.3 System specificity

The principal quantitative results for the evaluation are specific to the system and application upon which the testing was conducted. The anticipation is, however, that the comparative qualitative results will be valid across a wider range of problems, that extrapolation will be possible to similar applications in the future. If the extant testing is used as a basis for the

initial choices in future applications, a library of such results would incrementally be built up which could reduce the search space for a growing range of applications.

The results for the current system should have application for systems which employ background estimation and/or object segmentation and/or object classification, especially in a resource scarce environment. The results would also have some relevance outside the resource-scarce area.

The choice of extensive intuitive approximations within the evaluation process reduced evaluation resources required, but also limited the generality of the results somewhat.

7.2.4 Evaluation metrics

The interpretation method devised for evaluation of performance relied on comparison against human-generated ground truth. Statistics as to true positive results and ROC plots were used to assist interpretation of the system variants behaviours. It was found that these results required extensive further analysis, often on frame by frame basis, to evaluate the causes of unanticipated performance thoroughly.

This gave interesting results for individual methods (see 7.4) but significantly increased the human resource time for the evaluation. This will be partially due to this being the first application of the method (if a library of previous results is available, less work may be required to interpret the observed patterns) but storage of the analyses as textual interpretations is inconvenient in terms of reuse. Even where the analysis burden is reduced, the nature of the method requires the storage of extensive contextual information concerning assumptions and problem-specific factors which is not ideal for a clear, objective characterisation.

The decision to classify pedestrian groups as single entities in the testing, to parallel human intuitive perceptions, will have had a significant bearing on the results for these situations. This will limit the generality of the results somewhat, to applications where this corresponds to the 'event' of interest.

Appropriate metrics must be devised for evaluating any additional modules in further applications of the methodology. For example, to extend to evaluating alternative tracking approaches would require metrics to evaluate how long a correct track is maintained, how many times it is lost or false tracking occurs etc. Again, this is a consideration which applies primarily for the

early uses of the approach, as details of metrics used should be included in the documentation supporting each usage

7.2.5 Resource requirements

The resource requirement measures are currently of limited use due to the system dependency of such metrics. Again, if a library of such evaluations on different platforms grew, extrapolation of the results would be more meaningful. The significant error (see 7.3.1) during the design process which prevented transfer of the application to the final platform presents a reduction in the quality of the overall results.

7.2.6 Test data

The test data collected covered a wide range of environmental conditions and object behaviours. To give clearer results, requiring less frame-wise analysis, a greater decoupling of environmental and object attributes would have been useful. The current work attempted to examine the same test data to evaluate changes due to both, again to reduce the testing resource requirements. In the end, the additional work required in analysis to separate the effects and the degree of ambiguity this gave for the results suggests that investment of additional up-front resource allocation may be more efficient.

In all scientific investigation, it is common practise to investigate systems by holding all variables except one static while another varies. Here, this would be equivalent to, for example, comparing scenes with constant lighting/weather/activity levels, which differ only in the amount of occlusion at the objects (which themselves should exhibit similar speed/scale). The attempt to short-cut this process here, by allowing multiple variables to vary simultaneously followed by a multi-dimensional statistical analysis has given ambiguous results requiring the supplementary frame-wise analysis.

7.2.7 Assumption of non-combinatorial nature

A key assumption of the evaluation methodology is that the assessment approach for the performance of the individual modules is not sufficient to support an efficient overall system design. The evaluation process for the individual modules is traditionally by high-level hu-

man interpretation of visual output, which may not be appropriate for considering modules which feed results to subsequent automated processes.

This is supported by the actual results. A 'traditional' evaluation of the background representation process suggests that the *Mixture of Gaussians* approach gives the better approximation to the background scene than the *median average* method. However, evaluating the results of system variants using the two approaches reverses this evaluation.

This is partially due to the greater range of situations which can realistically be used in evaluation for the system overall. It is difficult to ascribe meaning to the emulation of a background representation process in varying conditions of occlusion, object size etc. It is only when the evaluation takes place within the context of the overall system that this sort of detailed consideration becomes meaningful. This is a significant strength of this approach to system evaluation.

7.2.8 Accessible storage of test data

The storage of the test data for use by third parties is an important aspect of the methodology. Ideally, it should be in the same format as that used directly as input by the test system, although the memory required for making the full data sets available is in the region of 4Gb. Compressed file formats were considered inappropriate due the possibility of information loss during the process and *.avi* files (from which the test image sequences were constructed) did not offer a significant memory usage saving.

The memory-intensive use of storage as a sequence of still *.ppm* images has been adopted in this pilot study [115], which may not be viable for larger databases. This is clearly not ideal, and the possibility of using the allegedly 'lossless' video compression offered by JPEG2000 is recommended as a future solution.

7.2.9 Conclusion on the design and evaluation methodology

As planned, the methodology presents a possible 'third way' for the evaluation of vision systems as part of the design process. The approach balances quality and generality of results with the resources applied to achieve them. The overall approach of evaluating composite system variants implemented from distinct modules is supported in that the results **do** diverge from those suggested by the traditional approach to evaluating individual modules.

As a 'pilot' exercise during which the methodology became more clearly defined and could be evaluated as a work in progress however, it does have clear limitations and areas for improvement. More decoupling of the controllable variables in the test data examples would allow for clearer statistical results with less need for supplementary analysis. If the methodology is adopted by other researchers, the library of test results would enhance the generality, but only if the results are made centrally available in some form with a clear taxonomy.

The approach is intended only as a medium-term solution in the absence of algorithm-level performance characterisation. Whether the methodology would represent an efficient use of resources in this context is questionable and depends to some degree on the speed and enthusiasm with which that is achieved.

An additional benefit of the methodology worth highlighting is in facilitating the integration of user specifications into the design/testing process. The process of validation of the initial design process with the user can be extended to cover not only the basic modular design, but also the selection of candidate solutions for test implementation. Also, where the user has specific demands in respects of certain scene types or operating conditions where the application must function well, they could specify the inclusion of representative data sets in the test phase, ideally proving real examples of such themselves.

A benefit which is of particular significance to the third generation distributed processing surveillance systems systems which are becoming more common is in allocation of processing load. The system architecture of such systems will often allow for processing tasks to be allocated dynamically between the (resource scarce) distributed processors and a (relatively resource rich) central processing unit or 'hub'. By altering the assumptions as to the relative importance of performance characteristics, different task allocation plans will be recommended as most appropriate by the method.

7.3 The overall results for the test design problem

7.3.1 Specification error

The most significant limitation in the quality of results for answering the current design problem originates from an error during the initial stage of the investigation. To give a quantitative confirmation of the comparatively evaluated system choice, that choice should have been im-

plemented directly on the VideoBridge^{TM} platform as the final stage in the process.

The test implementations for the system should have been written in code that was closely compatible with the language to be used on the final platform, so that transferral process would have involved only relatively minor modifications. In the event, the language used by the CamOS operating system was not investigated in sufficient detail, which error led to a divergence between the test implementations and the desired final result.

The performance evaluation had to be prematurely curtailed at the stage of the most appropriate system variant recommendation from test implementation results, without being able to evaluate **actual** resource demands on the final platform. Further, an option to generate code which may have been considerably faster to transfer to the final platform was not discovered until the very end of the write-up period.

Several valuable lessons have been reaffirmed as an outcome of this error. Establishing the system requirements is an elementary but central stage of any systems design problem, and the problems caused by a deficiency in this area here serve to underline its importance in future applications of the methodology. Time spent in a full, detailed evaluation of the problem and its context is seldom wasted and the need to question **all** assumptions and maintain good communication with the user is highlighted here.

Further, the abilities and limitations of the tools used (here the C++ and CamOS compliers particularly) should be explored in sufficient detail that their potential can be fully employed.

7.3.2 Timing and memory metrics

The timing and memory results were obtained using *UNIX* procedures for the overall processes and at intermediate points of operation. Their use could be refined in future evaluations to record minima, maxima, averages and variances for each value and to record modal and mean averages. Data on the spread of values would be an important additional information element for evaluation of processes.

It should be noted again that the results are platform specific and also that they can be affected by external considerations. Safeguards to minimise any distortion due to system use by multiple users are vital to mitigate theses platform effects.

7.3.3 Most appropriate selection

The recommendation for the most appropriate system variant within the context of projected use for the VideoBridge^{TM} system was not as anticipated. The most appropriate variant as selected in 6.6 is the median/model combination, using the boundary tracing object growing module.

This is interesting as the variant suggested by initial qualitative characteristics would be the Gauss/model combination. The secondary expectation was that the fully minimised system using the most economical approach to processing (the median/shape variant) could be more appropriate given the resource restrictions of the specific application.

The fact that the result diverges from both of these supports the general idea that a modular, combinatorial system variant evaluation is appropriate. Whether the result is in fact correct is more difficult to ascertain. A possibility to consider for obtaining this comfort outside the routine methodology proposed would be the re-implementation of all system variants for a test case onto the final platform. This would provide a valuable check both on the general conclusions and the comparative validity of the evaluation cross-platform.

7.3.4 Conclusion on the overall results for the test design problem

Following on from the assessment of the evaluation process itself, the recommendation for the current design problem is made with many provisos, limitations and assumptions. The recommendation only holds in a specific and limited application context which may not be adequate given the rate of development common in such systems.

The large amount of frame-wise analysis and subsequent qualitative description of effects is not ideal in an engineering context and brings an unwelcome element of further ambiguity to the results evaluation.

The most significant limitation to the current results comes from the inability to transfer the most appropriate variant cleanly to the final platform. Due to this, the results fall short of the ultimate aim of generating a working system solution. The importance of clearly and unambiguously understanding the system specifications in future evaluations cannot be stressed too greatly.

7.4 The performance results for individual modules

7.4.1 General

The results for individual modules flow from evaluation of system variants where the specific solution used for a particular module is the only significant variable. In the pilot application of the evaluation methodology, the individual module testing was restricted to establishing basic functionality and setting module-specific parameters to optimise the image-like output from a module. The results of this stage would be of more general use if an ROC curve for a range of parameter settings for each module implementation were recorded. This would allow informed selection of implementations for situations where the weighting applied to true/false results vary.

7.4.2 Background representation and segmentation

The median background representation approach was a straight re-implementation of previously published work in the area [28].

The *mixture of Gaussians* approach implemented differed from earlier versions [48] in that the restriction to fixed point arithmetic mandated that exact calculations of exponent values for probability were not possible. Examination of the algorithm revealed that these were not actually necessary for evaluating initial matches, providing an avenue for reducing computations in the approach generally. Approximation of the probability calculation for matched points was possible within the allowable range using a look-up table.

Although the *mixture of Gaussians* approach performs best in instances of median occlusion and scenes with a large degree of oscillatory motion generally, there were unexpected failure results in 'twilight' scenes with artificial lighting and low levels of background/foreground contrast (section 6.3.3). The current implementation also exhibited relatively high false positive detection rates across all scene types (table 6.5 and figures 6.14 and 6.15) and an increased time resource demand of the order of 100% (table 6.6).

These are interesting results that were not anticipated and resulted in the conclusion that, for this problem, the median background approach was to be preferred.

7.4.3 Object labelling

Both CCA approaches implemented for the system were too slow for effective use in a real-time system with limited resources. Fast CCAs do exist which may make the approach more feasible but comparing the relative dimensionality of the search space, these would still be slower than a similarly improved boundary-following approach. In terms of speed, this approach is far superior to the CCA method and is equal in effectiveness for object locating, moment calculation, pose estimation and outline specification.

The main drawback of the approach is that the precise size is not automatically generated. In the current system an ellipsoid approximation is suitable for potential pedestrians, but fast approaches to give the exact area in pixels are possible (using e.g. a 1-D horizontal count between boundary points).

7.4.4 Object classification

The shape-based classification approach is a novel implementation developed from first principles to be a fast and straightforward method to select pedestrian objects.

The model-based classification approach differed from earlier versions in that an alternative approach was taken to matching a slightly altered model. The models used consisted of the mean shape for an object, arrays of eigenvalues and eigenvectors corresponding to its observed characteristic distortions and a new element, the positional variance at each point on the object. The interpolation matrix used in the original implementation during the model matching process was eliminated in the new version.

In the revised version, the matching process was approximated as a linear optimisation problem in multiple variables, to be solved using the *Simplex Method*. This reduced the extent of the spline interpolation process required and generated results that, with the stored positional variances, were used to update simplified Kalman filter predictions for each point.

The revised version functioned well within the system, giving an improvement of approximately 11% in true positive identifications compared to the shape-based approach (section 6.3.2). Although proving the concept within this context, an evaluation against the original approach should be conducted as a separate exercise to evaluate their comparative performances.

Although the model-based approach gives a maximum false positive rate approximately 1%

higher than the shape-based approach and equates with a 6% increase in processing time per frame, the median/shape variant system encounters significant difficulties in scenes with a large amount of ground shadowing (section 6.3.2).

On balance, the results did not give a clear general 'winner' for this module but the fact that the range of results can be evaluated in terms of a specific problem is an advantage of the overall methodology. Within the context of the current problem, the shape-based approach gave the best all-round performance.

7.4.5 Conclusion on the performance results for individual modules

The level of results on individual modules within this methodology is limited to that of qualitative comparison between approaches. As such, the results serve as a starting point for more detailed quantitative comparison in areas of interest, as well as providing supplementary data to assist in choosing candidate solutions in later design problems.

It would be interesting to extend the module evaluations to further test data, to see if predictions based upon the current results were fulfilled.

7.5 Overall conclusions

The methodology proposed has been proven to the extent that it has been shown to be capable of producing unanticipated results for the relative abilities of a subset of system configuration options for a visual surveillance application. The approach has allowed selection of an appropriate system variant to be implemented on the final platform and has provided more general comparative performance and resource evaluation for the candidate approaches at the module level.

The results are generalisable in two main areas. First, to the extent that the methods employed have been qualitatively compared at the module level. Secondly, at the system level, where the quantitative results may be used to extrapolate performance predications for similar problems. By publishing the detailed quantitative results with supporting contextual information and analysis and making the test data available to third parties [115], the data can serve as the basis for an evolving library of such results. This library can be used as an improved source for initial data for future design problems.

The limitations of the approach are primarily a characteristic of its nature as a compromise solution between the full generalisation ultimately desired (from algorithm level performance characterisation across a range of platforms) and the resources required to achieve the results. The 'freezing' of individual module implementations, including tuning parameters based on the image-like output is likely to introduce sub-optimality in the overall system. Assessment of the intermediate output using a human evaluation is inevitably anthropocentric and does not factor in any consideration of error propagation along the system.

As an evaluation process which must provide practical performance characterisation results for a specific application/platform combination, the quantitative results generated are only strictly applicable to that case. The inferences which may be drawn to give evaluation useful in the general case for third parties is more qualitative, and provides comparisons only over a subset of algorithm combinations.

The results of the pilot application required extensive frame-wise re-analysis, making them less easily interpretable at a glance. Further, as with all such empirical testing approaches, it is the specific **implementation** of each algorithm and procedure which the final results relate to: if the programming quality is inferior, the results may be distorted.

A major restriction in this instance of the methodology's use was the error in the choice of test programming language. The primary testing was conducted on the local network for speed and convenience, but should have been supplemented with a final test suite while running on the final platform.

It terms of individual module implementations, it has been shown that a *mixture of Gaussians* background representation using only fixed point arithmetic is feasible. It was also shown that several efficiency savings were possible by removing certain probability calculations altogether. A fast object labelling approach, based on boundary following and shape approximation has been shown to be effective where a specific object type is of interest. A novel version of flexible model matching using a linear optimisation approximation and applying the *Simplex Method* has been presented and shown to function.

In comparing module performances, the most interesting result was that the *mixture of Gaussians* approach did not perform as well as the *median average* method over a wide range of scene types. Although it performed better with occlusion instances and scenes with oscillatory motion, the *mixture of Gaussians* approach performed quite poorly in scenes with low levels of

background/foreground contrast.

While the superior performance of the model-based object classification method over the shape based approach **was** as anticipated, the increased processing time for the whole system due to its use (of only 6%) was lower than anticipated and make it appear suitable even in resource scarce applications.

Overall, the project has presented a novel design and evaluation methodology which has been shown to be capable of producing novel and interesting results for modular systems based on new and extant approaches. The results from this pilot application can serve as useful data for future system design problems. If the method is further developed, to refine the balance between resources invested and results generated, then the quality and usefulness of the information generated to add to a growing data base can only increase.

> *This thesis proposed that an integrated design and characterisation methodology for the construction of vision systems for a specified application, based on object oriented design principles, could efficiently provide a recommendation as to an appropriate implementation. Such a methodology could also contribute to an evolving database of test sequences and comparative performance information of more general use to third parties.*

This has been shown to be the case, given the limitations discussed and the further development of the methodology.

7.6 Further work

The next step following on from this work in the short term should be the migration of the recommended *system variant* to the CamOS platform. This would provide both a practically useful deliverable application and corroborative evidence as to the quality of the results of the characterisation process.

Taking the lessons of the current 'pilot' implementation, the next step in the development of the design and evaluation methodology should be to generate further results using more decoupled scene data. New implementations of the modules which are compatible with the final platform should be used for the current design problem or as appropriate for a new one.

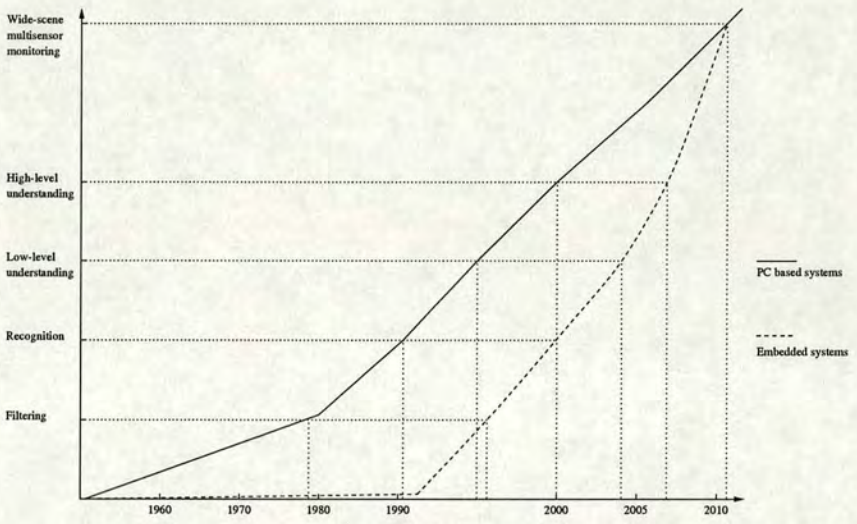

Figure 7.1: *Technical evolution of multimedia surveillance systems versus time*

If the current design problem were re-examined, the results from this second application should be evaluated in tandem with the current results and analyses. Although less 'frame-wise' analysis will be required, the results should provide additional comfort in respect of such analyses made in this project.

More detailed results on the performance of individual modules could be recorded and both additional analysis of system performance and extension modules considered. The additional analysis could involve evaluation of tracking performance with extensions including a full implementation of face capture/association with general scene footage.

7.7 The future

7.7.1 Conclusions in context

To put the overall conclusions into context to assess their potential value, one need only consider the continually accelerating pace of technological advances in the areas of surveillance and of computing generally. Computing power available from a single chip has historically grown according to a broadly exponential curve, and figure 7.1, taken from [117] illustrates the technical evolution of video-based surveillance systems versus time in the civil field, with extrapolation to the end of the decade. The processing power offered by the new releases of IndigoVision's technology as compared with the releases extant at the start of this project has

already increased significantly and with continuing advances the resource availability aspects of the problem will continue to vary.

The continuing growth in the capabilities of mobile communications and computing units offers whole new areas for increasingly flexible image sequence processing and interpretation. Dynamic reconfigurability of distributed and mobile units by down-loading software tools on demand are anticipated to be industrially feasible in the next decade, as is the advent of ad hoc wireless networks for public security aims [117]. With the growth of such markets, the already extensive research effort focused on advanced surveillance development is likely to intensify.

In such an environment, the efficient design and evaluation of appropriate vision applications over acceptable time-scales and which fulfil varied user requirements will become of even greater importance. Algorithm-level performance characterisation is only likely to become more prevalent gradually within the vision community: the change is of the nature of a paradigm shift in terms of the design and evaluation approach.

Given this, the natural tendency would be to continue with the traditional monolithic approach to design, almost entirely focused on the single application under development, with no additional resource made available for providing more generalisable data for subsequent tasks. With the increasing range of potential solutions in the literature from which to select candidate solutions, achieving a near-optimal result in this environment will become ever less likely.

A refined version of the presented methodology constitutes one approach to providing a more flexible approach to the design problem incorporating limited comparative testing. With the accelerated pace of development envisioned would come increased pressure for timely and effective design. The ability to 'test run' a subset of candidate solutions, with value added in the form of incrementally generating an accessible database of test data and qualitative performance comparison information is an attractive proposition.

A methodology which addresses the new *systems architecture* issues presented by distributed/mobile technologies to give a quantitative evaluation of alternative task distributions is presented in [118] and would constitute an appropriate companion evaluative method.

7.7.2 Broader surveillance issues

Surveillance in some form is becoming an increasing part of everyday life in a variety of environments. Whether implementing an automated 'Neighbourhood Watch' to protect the properties of Middle England, identifying vandals in the inner cities, implementing the forces of government oppression to discourage free expression or just allowing more effective monitoring of contestants in the latest 'Reality TV' game show, routine surveillance is essentially a *fait accomplis*.

The best that can realistically be hoped for from an engineering design/evaluation methodology is that the closet to the optimum implementation is recommended for the final implementation. This should in turn result in the provision of the highest quality results for a given system, so that negative effects due to problems with the technology itself may be minimised.

Whether the overall effects of the ubiquitous nature of the technology on society are to the good or ill really depend, as with so many aspects of technology, on the use to which they are put. No evaluation methodology from a technical standpoint can be expected to take quite so global a view of performance as to formally incorporate concerns at this level.

In final summary, surveillance technology seems certain to become an increasing part of contemporary society in ever broader areas. The attitudes to this have changed almost beyond recognition since the visions of Orwell's dystopia. The presented methodology answers the requirement for an improved technical evaluation strategy for such systems. An effective evaluation of applications of the technology in social and ethical terms, though more the province of philosophy, sociology and privacy law, would be an ideal partner in the future. To paraphrase the words of the **new** Big Brother (figure 7.2) "N solutions. One Winner. You Decide."

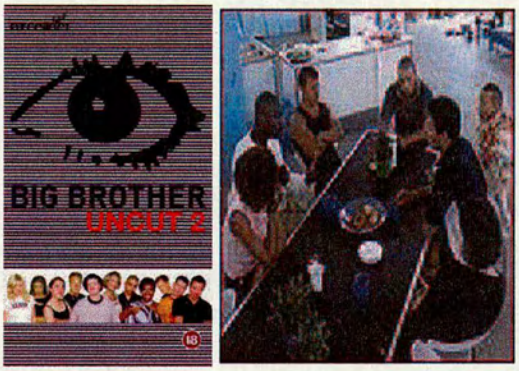

Figure 7.2: *Ten Contestants. One Winner. You Decide.*

Appendix A
Class diagrams

This appendix provides additional detail with respect to the Object Oriented class structure aimed for in the implementation of the modules of the vision processing system. The development was not always strictly in line with the deals of OO design and some further work to achieve the structural goals is still desirable.

The class diagrams are abbreviated versions of UML class diagrams used in planning and design of C++ applications commercially. The principle purpose of the diagrams is to specify the data and process attributes of each class and to indicate inheritance and association relationships between them.

The diagrams used herein are simplified in two main ways. The processes within a class are listed in the lower half of a rectangle corresponding to that class (as is usual), but normally the arguments of these operations are specified as well. As a full documentation of the class is not needed here, it is sufficient to give an indicative list of the member processes by name.

Also, the access functions required to get or set the private data members are not listed explicitly except, as an example, in figure A.1. All of the classes have such functions which must be used to access these data members.

Figure A.1 illustrates the inheritance relationship between the VS_frame and VS_image and corresponding input/output classes. The VS_frame class is the basic image unit of the Department of Electrical Engineering's Vision Systems Library, and was written independently by fellow group members.

The VS_image class is the basic image unit written at the outset of the project and was retrospectively modified to make it a descendent of the VS_frame class to achieve compatibility with the Vision Systems Library.

The VS_point, VS_win and VS_nhood8 classes illustrated in figure A.2 are supplementary classes written alongside VS_image to provide easier specification of pixel positions, rectangular windows and pixel neighbourhoods.

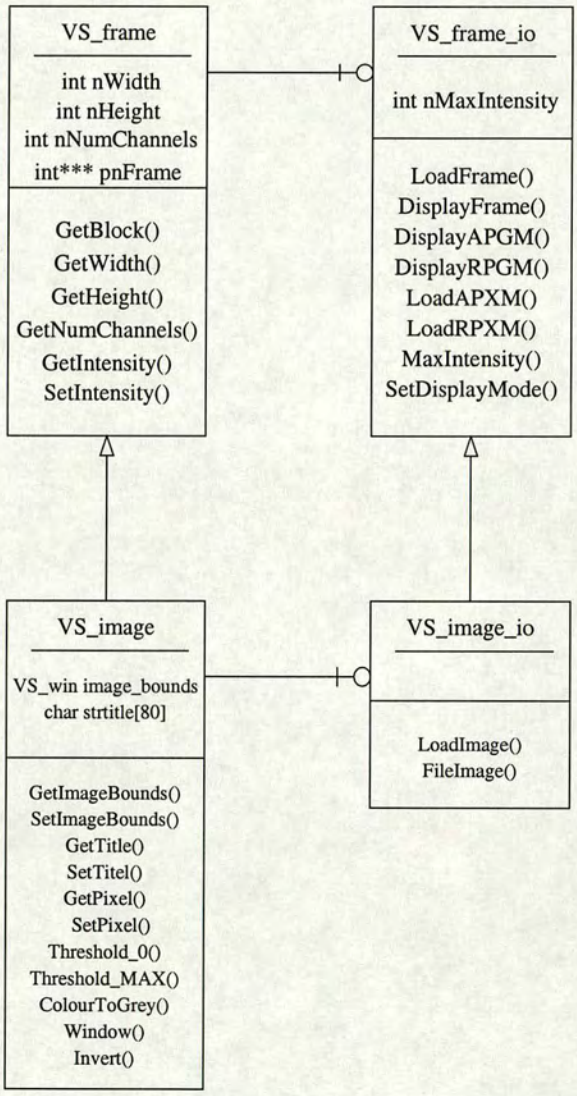

Figure A.1: *VS_frame and VS_image and corresponding input/output classes.*

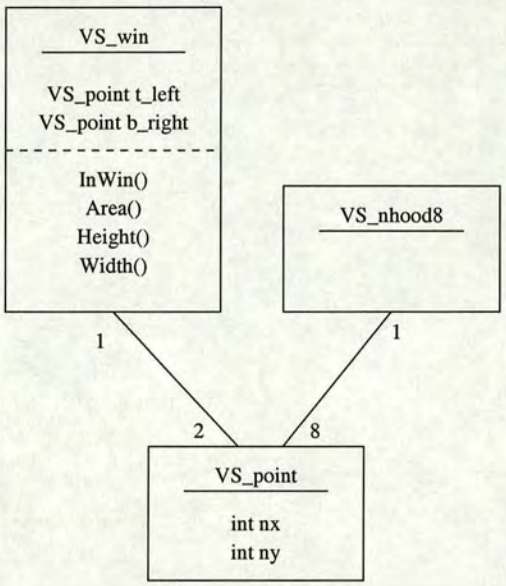

Figure A.2: *VS_point, VS_win and VS_nhood8 classes.*

The VS_image_process class is another which was written independently by fellow group members as part of the Vision Systems Library. It provides a virtual function (*Process()*) which can be overwritten by derived classes and is useful for implementing the simple vision processes as illustrated in figureA.3.

VS_tracker is the top-level class implementing the overall vision system and has two distinct versions (as illustrated in figure A.4) corresponding to the choice of object classification approach. In addition to the elements shown in that figure, the class contains objects from the other modules as indicated in figure A.5.

A class exists to define objects for each module, which usually have two alternative descendents corresponding to different *candidate solutions*. Figure A.6 illustrates the distinction between the two background approaches in this respect: VS_median is a descendent of VS_image_process (see figure A.3) whereas VS_mix_back is much more complex and also contains objects of class VS_gauss_back (which themselves contain objects of class VS_gaussian). Figure A.7 shows the distinction between the classes corresponding to CCA labelling and the boundary following approach. Note that, by standard OO design conventions, the common elements from the two should be moved to the parent *VS_labeller class* for efficiency.

Figure A.8 illustrates the most complex module in terms of OO class relationships, the object

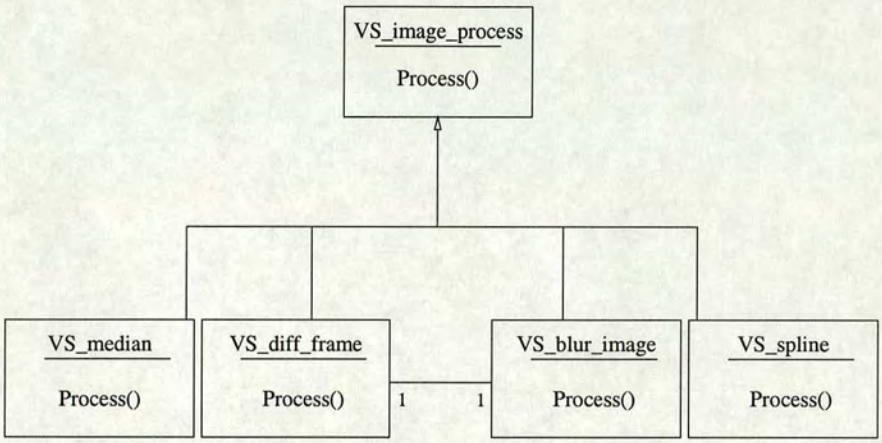

Figure A.3: *VS_image_process descendent classes.*

classification stage. The two classifier derived classes each have unique classes providing objects, but both require a VS_outline object for boundary extraction. It should be noted that many of the elements of the VS_outline class should be relocated into a separate class for bound box building and evaluation which are only required by VS_shape_class objects.

To achieve the most elegant and clear structure, it would also be desirable to transfer some functionality in respect of model matching and tracking between the classifier and tracker classes. Finally, the VS_shape and VS_pedestrian classes should derive from a supper class which collects their numerous common attributes.

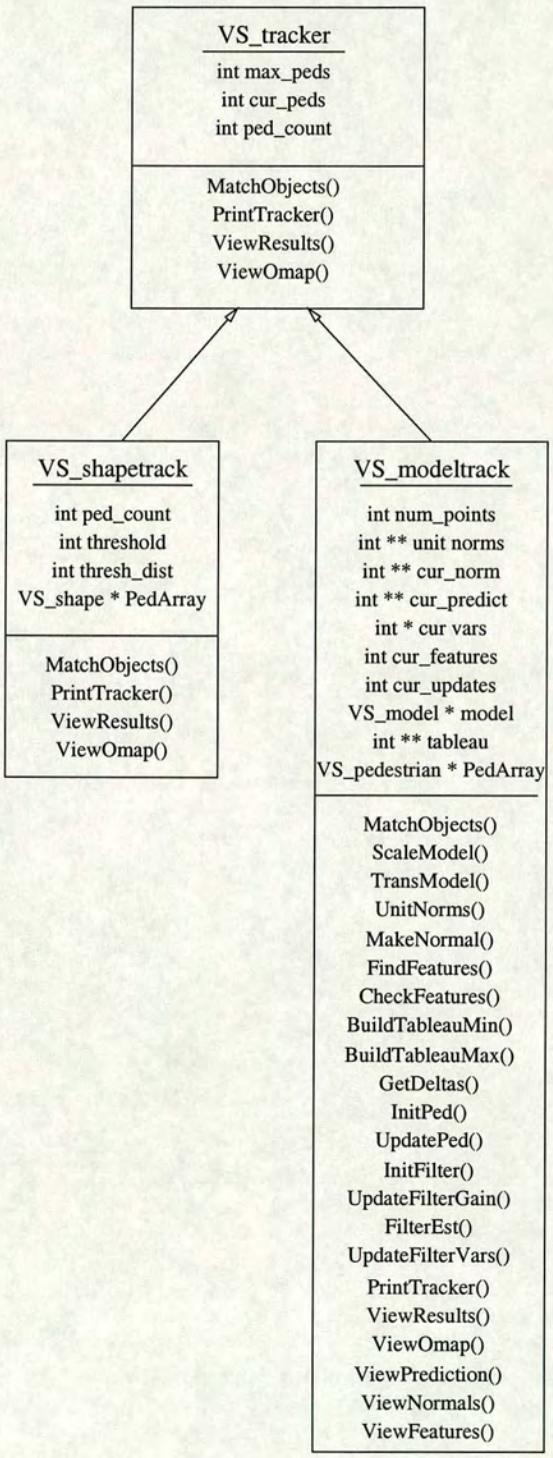

Figure A.4: *VS_tracker descendents corresponding to object classification module choice.*

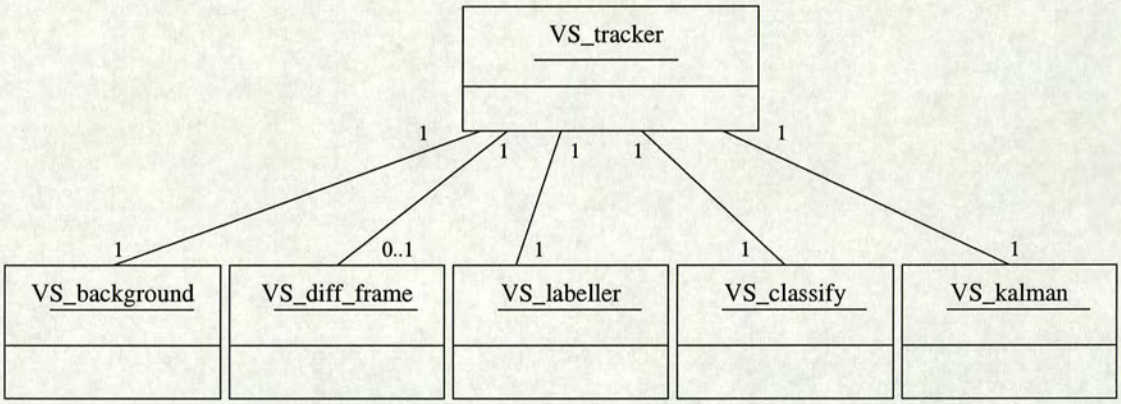

Figure A.5: *Tracker analysed by module super-classes.*

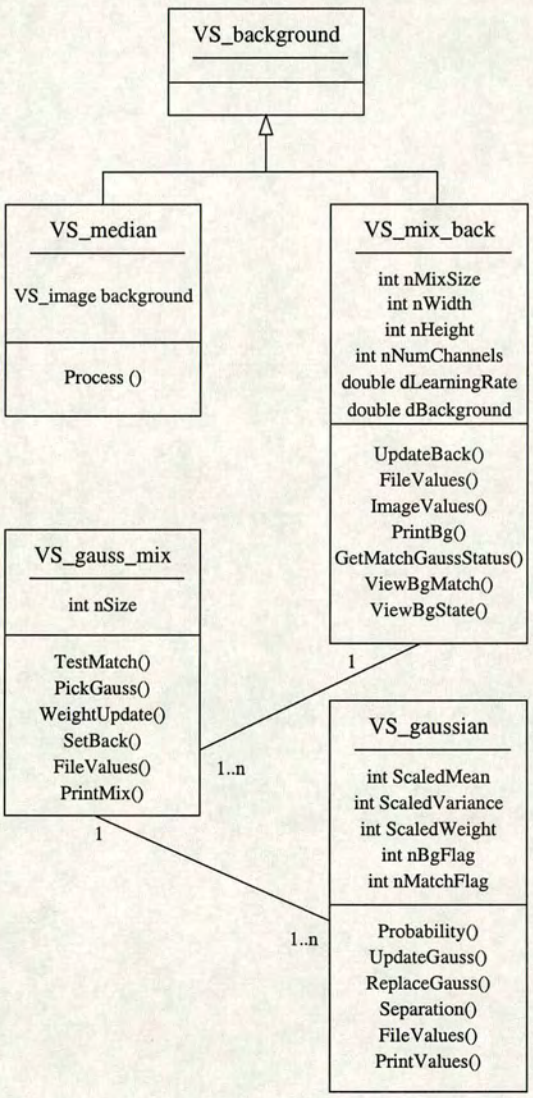

Figure A.6: *Background generation module classes.*

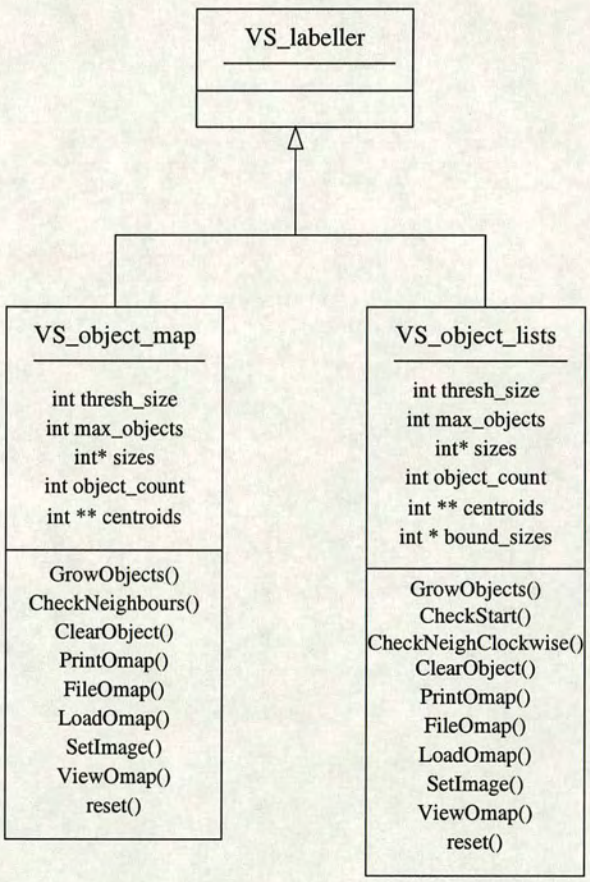

Figure A.7: *Object labelling module classes.*

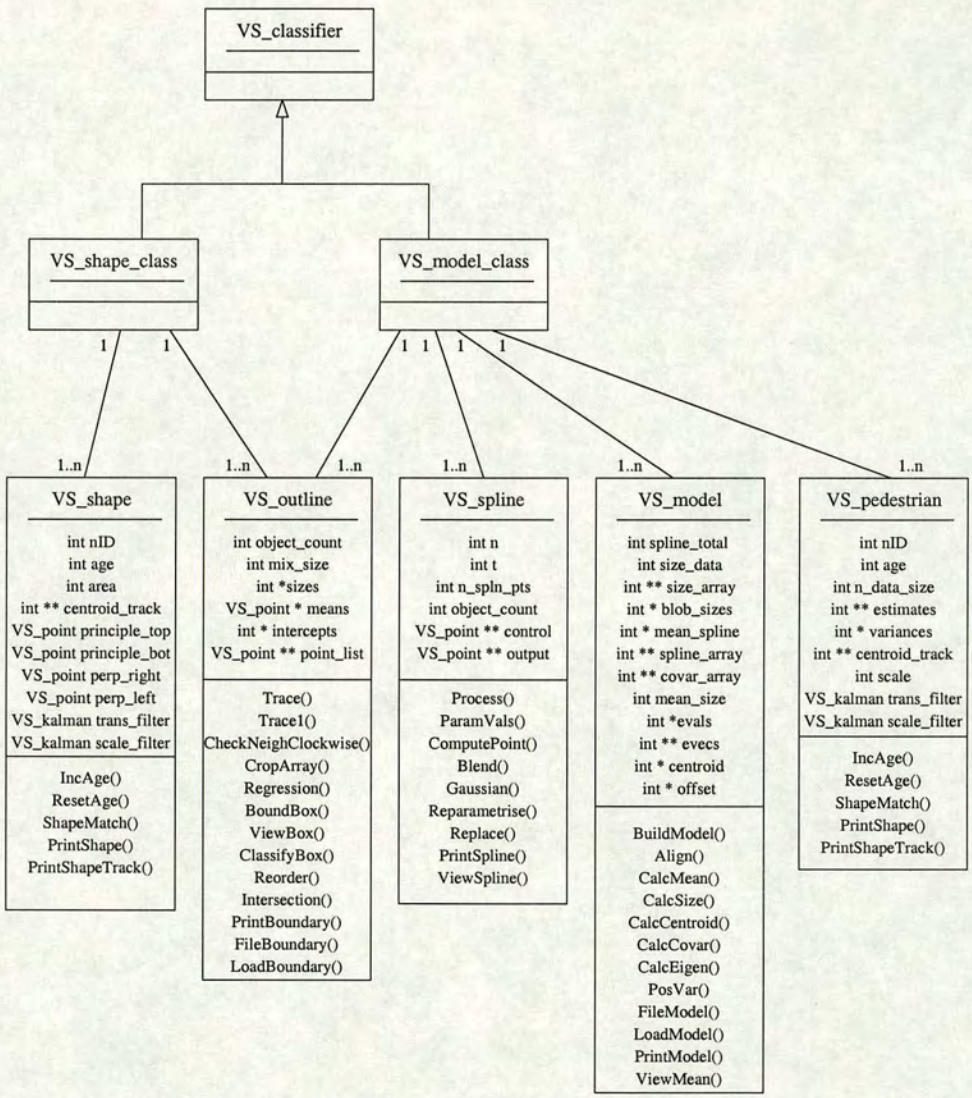

Figure A.8: *Object classifier module classes.*

Appendix B
Pedestrian Sequence Events Sheet

Pedestrian Sequence Events Sheet			
Scene:			
	Time	**Frame**	**Event Description**
Start			
End			
Start			
End			
Start			
End			
Start			
End			
Start			
End			
Start			
End			
Start			
End			
Start			
End			
Start			
End			
Start			
End			
Start			
End			
Start			
End			
Start			
End			

Figure B.1: *Example of format for Pedestrian Sequence Events Sheets*

Bibliography

[1] W. Press, S. Teukolsky, W. Vetterling, and B. Flannery, *Numerical Recipes in C : The Art of Scientific Computing*. Cambridge, 1992.

[2] K. Vertanen, "http://www.bookcase.com/library/software/msdos.devel.lang.c++.html," 1994.

[3] G. Orwell, *1984*. Secker & Warburg, 1949.

[4] A. M. Burton, S. Wilson, M. Cowan, and V. Bruce, "Face recognition in poor quality video: evidence from security surveillance," *Psychological Science*, vol. 10, pp. 243 – 248, 1999.

[5] S. Davies, "Something to watch over us," *BBC Online Network*, May 15 1999.

[6] J. Griffiths, *Market Research Report on CCTV*. Key Note Publishers, 2000.

[7] P. International, "www.privacyinternational.org," 2001.

[8] C. Milner, "Secret spies keep watch on our streets," *Electronic Telegraph*, vol. 1, May 13 1999.

[9] S. Davies, "Spy cameras don't beat street crime," *Electronic Telegraph*, July 23 1999.

[10] M. Foucault, *Discipline and Punish: The Birth of the Prison*. Pantheon Books, 1977.

[11] D. R. Davies and R. Parasuraman, *The Psychology of Vigilance*. Academic Press, 1982.

[12] N. H. Mackworth, "The breakdown of vigilance during prolonged visual search," *Quarterly Journal of Experimantal Psychology*, vol. 1, pp. 6 – 21, 1948.

[13] N. Wright, A. McGown, and J. Montgomery, "Relationships between brain activity and cognitive performance during overnight work: prediction of errors," in *Proc. of the 2nd International Conference on Methods and Techniques in Behavioral Research*, (Groningen), pp. 18–21, August 1998.

[14] D. Marr, "Early processing of visual information," *Phil. Trans. Royal Soc. London*, pp. 97–137, 1976.

[15] H. de Garis and M. Korkin, "Using evolvable hardware techniques to build a 75 million neuron artificial brain to control the many behaviors of a kitten robot," *Handbook of Biomimetics*, pp. 729–736, Sept. 2000.

[16] I. Pavlidis and V. Morellas, "Two examples of indoor and outdoor surveillance systems: motivation, design and testing," in *Proc. 2nd European Workshop on Advanced Video-Based surveillance Systems*, pp. 285 – 296, Kluwer, Sept. 2001. Kingston.

[17] R. Jain, R. Kasturi, and B. Schunck, *Machine Vision*. McGraw-Hill, 1995.

[18] K. W. Bowyer, "Experiences with empirical evaluation of computer vision algorithms," in *Performance Characterization in Computer Vision*, pp. 3–16, Kluwer, 2000.

[19] P. Courtney, "Evaluation and validation of computer vision algorithms," in *Performance Characterization in Computer Vision*, pp. 17–28, Kluwer, 2000.

[20] P. Courtney and N. Thacker, *Tutorial on Performance Characterisation of Computer Vision Techniques*. ECCV, 2000.

[21] P. Phillips, H. Moon, P. Rauss, and S. Rizvi, "The feret evaluation methodology for face-recognition algorithms," in *Proc. Computer Vision and Pattern Recognition*, pp. 137–143, 1997.

[22] A. Clark and P. Courtney, "Databases for performance characterization," in *Performance Characterization in Computer Vision*, pp. 29–40, Kluwer, 2000.

[23] R. Marik, "Quality in computer vision," in *Performance Characterization in Computer Vision*, pp. 41–51, Kluwer, 2000.

[24] A. Teschioni and C. S. Regazzoni, "Performances evaluation strategies for an image processing systems for surveillance applications," in *Proc. Workshop on Advanced Video Based Surveillance*, (Genoa), pp. 76 – 90, Kluwer, April 1998.

[25] P. Remagnino, A. Baumberg, T. Grove, D. Hogg, T. Tan, A. Worrall, and K. Baker, "An integrated traffic and pedestrian model-based vision system," in *Proc. British Machine Vision Conference*, pp. 380–389, BMVA, Sept. 1997.

[26] A. Baumberg and D. Hogg, "Learning flexible models from image sequences," in *Proc. European Conference on Computer Vision*, vol. 1, pp. 299–308, May 1994.

[27] A. Baumberg and D. Hogg, "An adaptive eigenshape model," in *Proc. British Machine Vision Conference*, vol. 1, pp. 87–96, BMVA, Sept. 1995.

[28] N. J. B. Mcfarlane and C. P. Schofield, "Segmentation and tracking of piglets in images," *Machine Vision and Applications*, vol. 8, pp. 187–193, 1995.

[29] E. Brookner, *Tracking and Kalman Filtering Made Easy*. Wiley, 1998.

[30] A. Baumberg, "Hierarchical shape fitting using an iterated linear filter," in *British Machine Vision Conference*, vol. 1, pp. 313–323, BMVA, Sept. 1996.

[31] A. Baumberg and D. Hogg, "An efficient method for contour tracking using active shape models," in *Proc. Workshop on Motion of Non-rigid and Articulated Objects*, pp. 194–199, IEEE Computer Society Press, Nov. 1994.

[32] N. Johnson and D. Hogg, "Learning the distribution of object trajectories for event recognition," *Image and Vision Computing*, vol. 14, pp. 609–615, August 1996.

[33] R. J. Morris and D. C. Hogg, "Statistical models of object interaction," in *Proc. ICCV '98 Workshop on Visual Surveillance*, pp. 81–85, 1998.

[34] A. Baumberg and D. Hogg, "Generating spatiotemporal models from training examples," *Image and Vision Computing*, vol. 14, pp. 525–532, June 1996.

[35] S. DeMa, "A self-calibration technique for active vision systems," *IEEE Transactions on Robotics and Automation*, vol. 12, no. 1, pp. 114–120, 1996.

[36] R. Hartley, "Self-calibration of stationary cameras," *International Journal of Computer Vision*, vol. 22, no. 1, pp. 5–23, 1997.

[37] L. de Agapito, E. Hayman, and I. Reid, "Self-calibration of a rotating camera with varying intrinsic parameters," in *Proc. British Machine Vision Conference*, BMVC, Sept. 1998.

[38] G. P. Stein, "Tracking from multiple view points: self-calibration of space and time," in *Proc. Image Understanding Workshop*, pp. 1037–1042, Nov 1998.

[39] D. Huts, J. Mazy, and K. Graf, "The prevention of vandalism in metro stations: Users requirements for advanced video-based solutions," in *Proc. Workshop on Advanced Video Based Surveillance*, (Genoa), pp. 34 – 43, Kluwer, April 1998.

[40] A. Pozzobon, V. Recagno, and G. Sciutto, "Security in ports: the user requirements for surveillance systems," in *Proc. Workshop on Advanced Video Based Surveillance*, (Genoa), pp. 18 – 26, Kluwer, April 1998.

[41] C. Sacchi, C. S. Regazzoni, and C. Dambra, "Use of video advanced surveillance and communication technologies for remote monitoring of protected sites," in *Proc. Workshop on Advanced Video Based Surveillance*, (Genoa), pp. 154–164, Kluwer, April 1998.

[42] C. S. Regazzoni, C. Sacchi, and E. Stringa, "Remote detection of abandoned objects in unattended railway stations by using DS/CDMA video-surveillance systems," in *Proc. Workshop on Advanced Video Based Surveillance*, (Genoa), pp. 165–178, Kluwer, April 1998.

[43] C. Nwagboso, "User focused surveillance systems for intelligent transport systems," in *Proc. Workshop on Advanced Video Based Surveillance*, (Genoa), pp. 8 – 17, Kluwer, April 1998.

[44] J. Renno, M. Tunnicliffe, G. Jones, and D. Parish, "Simulation of a video surveillance network using remote intelligent security cameras," in *Proc. IEEE International Conference on Networking*, (France), pp. 766–775, Springer Verlag, July 2001.

[45] S. Russel and S. Zilberstein, "Composing real-time systems," in *Proc. International Joint Conference on Artificial Intelligence*, pp. 212–217, 1991.

[46] P. M. Ngan, "Motion detection in temporal clutter," in *Proc. Third Asian Conference on Computer Vision*, pp. 615–622, Jan 1998. Hong Kong.

[47] G. Andolfi, M. Aste, M. Boninsegna, R. Cattoni, A. Potrich, and B. Caprile, "The advanced visual monitoring project at IRST," in *Proc. Workshop on Advanced Video Based Surveillance*, (Genoa), pp. 130 – 139, Kluwer, April 1998.

[48] C. Stauffer and W. Grimson, "Adaptive background mixture models for real-time tracking," in *Proc. Conference on Computer Vision and Pattern Recognition*, pp. 246–252, IEEE, June 1999.

[49] P. KaewTraKulPong and R. Bowden, "An improved adaptive background mixture model for real-time tracking with shadow detection," in *Proc. 2nd European Workshop on Advanced Video-Based surveillance Systems*, pp. 149 – 158, Kluwer, Sept. 2001. Kingston.

[50] N. Friedman and S. Russell, "Image segmentation in video sequences: a probabilistic approach," in *Proc. of the Thirteenth Conference on Uncertainty in Artificial Intelligence*, (Providence), pp. 175–181, Morgan Kaufmann, May 1997.

[51] T. E. Boult, R. Micheals, X. Gao, P. Lewis, C. Power, W. Yin, and A. Erkan, "Frame-rate omnidirectional surveillance and tracking of camouflaged and occluded targets," in *Second IEEE International Workshop on Visual Surveillance*, pp. 48–55, 1999.

[52] M. Xu and T. Ellis, "Colour-invariant motion detection under fast illumination changes," in *Proc. 2nd European Workshop on Advanced Video-Based surveillance Systems*, (Kingston), pp. 335 – 345, Kluwer, Sept. 2001.

[53] X. Gao, T.E.Boult, F. Coetzee, and V. Ramesh, "Error analysis of background adaption," in *Proc. Conference on Computer Vision and Pattern Recognition*, pp. 503–510, IEEE Computer Society, IEEE Computer Society, 2000.

[54] Y. Kuno, "Detecting and tracking people in complex scenes," in *Proc. 2nd European Workshop on Advanced Video-Based surveillance Systems*, (Kingston), pp. 3 – 15, Kluwer, Sept. 2001.

[55] M. Kass, A. Witkin, and D. Terzopoulos, "Snakes: active contour models," in *Proc. 1st International Conferance on Computer Vision*, (London), pp. 259–268, 1987.

[56] T. Cootes and C. Taylor, "Active shape models - 'smart snakes'," in *Proc. British Machine Vision Conference*, pp. 276–285, BMVA, September 1992.

[57] T. Cootes, C. Taylor, D. Cooper, and J. Graham, "Training models of shape from sets of examples," in *Proc. British Machine Vision Conference*, pp. 9–18, BMVA, September 1992.

[58] N. Peterfreund, "The velocity snake," in *Proc. Workshop on Nonrigid and Articulated Motion Workshop*, (Puerto Rico), pp. 580–591, IEEE, June 1997.

[59] N. Peterfreund, "Robust tracking of position and velocity with kalman snakes," *Transactions on Pattern Analysis and Machine Intelligence*, vol. 21, pp. 564–569, June 1999.

[60] R. P. Wildes, "A measure of motion salience for surveillance applications," in *Proc. International Conference on Image Processing*, (Chicago), pp. 183–187, IEEE, October 1998.

[61] S. Huwer and H. Niemann, "Adaptive change detection for real-time surveillance applications," in *Proc. 3rd IEEE Workshop on Video Surveillance*, (Dublin), pp. 37 – 45, IEEE Computer Society, July 2000.

[62] N. Kehtarnavaz and F. Rajkotwala, "Real-time vision-based detection of waiting pedestrians," *Journal of Real-Time Imaging*, pp. 433 – 440, 1997.

[63] C. Wren, A. Azarbayejani, T. Darrell, and A. Pentland, "Pfinder: real-time tracking of the human body," *Transactions on Pattern Analysis and Machine Intelligence*, vol. 19, pp. 780–785, July 1997.

[64] J. Orwell, P. Remagnino, and G. Jones, "From connected components to object sequences," in *Proc. 1st IEEE Workshop on Performance Evaluation of Tracking and Surveillance*, pp. 72 – 79, 2000.

[65] A. Mohan, C. Papageorgiou, and T. Poggio, "Example-based object detection in images by components," *Transactions on Pattern Analysis and Machine Intelligence*, vol. 23, pp. 349–361, April 2001.

[66] Y. Nieverget, *Wavelets Made Easy*. Birkhauser, 1999.

[67] V. Vapnik, *Statistical Learning Theory*. Wiley, 1998.

[68] S. M. Smith, "Asset-2: Real-time motion segmentation and shape tracking," Tech. Rep. TR95SMS2b, Department of Clinical Neurology, Oxford University, 1995.

[69] G. Halevi and D. Weinshall, "Motion of disturbances," in *Proc. Conference on Computer Vision and Pattern Recognition*, pp. 897–902, IEEE, IEEE, June 1997.

[70] R. Rosales and S. Sclaroff, "Improved tracking of multiple humans with trajectory prediction and occlusion modeling," in *Proc. Conferance on Computer Vision and Pattern Recognition, Workshop on the Interprtation of Visual Motion*, (Santa Barbara), IEEE, IEEE, June 1998.

[71] Q. Cai and J. K. Aggarwal, "Tracking human motion in structured environments using a distributed-camera system," *Transactions on Pattern Analysis and Machine Intelligence*, vol. 21, pp. 1241–1247, Nov 1999.

[72] A. Anzalone and A. Machi, "Video-based management of traffic light at pedestrian road crossing," in *Proc. Workshop on Advanced Video Based Surveillance*, (Genoa), pp. 49 – 57, Kluwer, April 1998.

[73] J. Krumm, S. Harris, B. Meyers, B. Barry, M. Hale, and S. Shafer, "Multi-camera multi-person tracking for easyliving," in *Proc. 3rd IEEE Workshop on Video Surveillance*, (Dublin), pp. 3 – 10, IEEE Computer Society, July 2000.

[74] A. Bakowski and G. Jones, "Video surveillance tracking using colour adjacency graphs," in *Proc. Conference on Image Processing and its Applications*, pp. 794–798, IEEE, June 1999.

[75] I. Haritaoglu, D. Harwood, and L. Davis., "W4: Who, when, where, what: A real time system for detecting and tracking people," in *Proc. Third Face and Gesture Recognition Conference*, pp. 222–227, April 1998.

[76] S. Ju, M. Black, and Y. Yacoob, "Cardboard people: a parametrised model of articulated image motion," in *Proc. 2nd International Conference on Face and Gesture Analysis*, (Vermont), pp. 38–44, 1996.

[77] I. Haritaoglu, D. Harwood, and L. Davis., "W4s: A real time system for detecting and tracking people in 2.5d," in *Proc. European Conference on Computer Vision*, (Frieberg), 1998.

[78] S. Sclaroff and J. Isidoro, "Active blobs," in *Proc. 6th International Conference on Computer Vision*, (Bombay), pp. 1146 – 1153, 1998.

[79] L.-Q. Xu and D. Hogg, "Neural networks in human motion tracking - an experimental study," *Image and Vision Computing*, vol. 15, pp. 607–015, 1997.

[80] B. A. Boghossian and S. A. Velastin, "Image processing system for pedestrian monitoring using neural classification of normal motion patterns," *Measurement and Control*, vol. 32, no. 9, pp. 261–264, 1999.

[81] J. Owens and A. Hunter, "Application of the self-organizing map to trajectory classification," in *Proc. 3rd IEEE Workshop on Video Surveillance*, (Dublin), pp. 77 – 83, IEEE Computer Society, July 2000.

[82] S. Gong and H. Buxton, "Visual observation as reactive learning," in *Proc. SPIE'92 International Conference on Adaptive and Learning Systems*, (Orlando, Florida), April 1992.

[83] P. Remagnino, G. Jones, and N. Monekosso, "Reasoning about dynamic scenes using autonomous agents," in *Proc. 7th Conference of the Italian Association for Artificial Intelligence*, (Bari), Springer Verlag, Sept 2001.

[84] H. Buxton and S. Gong, "Advanced visual surveillance using bayesian networks," in *Proc. International Conference on Computer Vision*, IEEE Computer Society, June 1995.

[85] K. Howarth and H. Buxton, "Visual surveilllance monitoring and watching," in *Proc. European Conference on Computer Vision*, pp. 321–334, Cambridge, 1996.

[86] P. Remagnino, T. Tan, and K. Baker, "Agent orientated annotation in model based visual surveillance," in *Proc. International Conference on Computer Vision*, (Bombay), pp. 857–862, ICCV, Jan. 1998.

[87] N. Chleq, F. Bremond, and M. Thonnat, "Image understanding for prevention of vandalism in metro stations," in *Proc. Workshop on Advanced Video Based Surveillance*, pp. 106–117, Kluwer, April 1998. Genoa.

[88] N. Rota and M. Thonnat, "Video sequence interpretation for visual surveillance," in *Proc. 3rd IEEE Workshop on Video Surveillance*, (Dublin), pp. 59 – 67, IEEE Computer Society, July 2000.

[89] D. Ayers and M. Shah, "Monitoring human behaviour from video taken in an office environment," *Image and Vision Computing*, pp. 833–846, 2001.

[90] E. B. Koller, "Modeling and recognition of human actions using a stochastic approach," in *Proc. 2nd European Workshop on Advanced Video-Based surveillance Systems*, (Kingstaon), pp. 17 – 28, Kluwer, 2001.

[91] S. Dettmer, A. Seetharamaiah, L. Wang, and M. Shah, "Model-based approach for recognizing human activities from video sequences," in *Proc. Workshop on Recent Advances in Computer Vision*, pp. 50–59, NSF, 1998.

[92] L. Davis, S. Fejes, D. Harwood, Y. Yacoob, I.Hariatoglu, and M.J.Black, "Visual surveillance of human activity," in *Proc. Asian Conference on Computer Vision*, pp. 267–274, 1998.

[93] R. Rosales, "Recognition of human action using moment-based features," Tech. Rep. BU 98-020, Boston University, Computer Science Department, Nov. 1998.

[94] C. Davies, J. Yin, and S. Velastin, "Crowd monitoring using image processing," *IEEE Electronics and Communication Engineering Journal*, pp. 37–47, 1995.

[95] C. Kaas, J. Luettin, R. Mattone, and K. Zahn, "Evaluation of a self-learning event detector," in *Proc. 2nd European Workshop on Advanced Video-Based surveillance Systems*, (Kingston), pp. 115 – 123, Kluwer, Sept. 2001.

[96] N. C. Haering, R. J. Qian, and M. I. Sezan, "Detecting hunts in wildlife videos," in *Proc. International Conference on Multimedia Computing and Systems*, vol. 1, pp. 905–909, IEEE, 1999.

[97] I. Limited, "http://www.indigo-avs.com/technology/indigo_univid.html," 2001.

[98] C. S. Regazzoni, G. Fabri, and G. Vernazza, eds., *Advanced Video-Based Surveillance Systems*. Kluwer, 1999.

[99] A. Peacock, S. Matsunga, D. Renshaw, and J. H. andA. Murray, "Reference block updating when tracking with the block matching algorithm," *IEE Electronic Letters*, vol. 36, pp. 309 –310, Feb. 2000.

[100] C. Haworth, A. Peacock, and D. Renshaw, "Performance of reference block updating techniques when tracking with the block matching algorithm," in *Proc. of the International Conference on Image Processing*, (Thessaloniki), October 2001.

[101] O. Firschein and T. Strat, eds., *Computer Vision Performance Characterization*. Morgan Kaufmann, 1997.

[102] P. K. Varshney and I. L. Coman, "Distributed multi-sensor surveillance: issues and recent advances," in *Proc. 2nd European Workshop on Advanced Video-Based surveillance Systems*, (Kingston), pp. 41 – 52, Kluwer, Sept. 2001.

[103] P. Courtney and N. Thacker, *Performance Characterisation In Computer Vision: The Role Of Statistics In Testing And Design*. Cambridge, 1999.

[104] C. Bishop, *Neural Networks for Pattern Recognition*. Clarendon Press, 1995.

[105] D. Hogg, N. Johnson, R. Morris, A. Buesching, and A. Galata, "Visual models of interaction," in *Proc. Leeds Annual Statistical Research Workshop*, vol. 18, pp. 97–100, July 1999.

[106] R.Bartels, J. Beatty, and B. Barsky, *An Introduction to Splines for Use in Computer Graphics and Geometric Modeling*. Morgan Kaufman, 1987.

[107] M. Robinson, "Did the earth move for you?: a novel metric for matching iconic images," tech. rep., MSc Report, Department of Artificial Intelligence, University of Edinburgh, June 1998.

[108] H. Buxton and S. Gong, "Visual surveillance in a dynamic and uncertain world," *Artificial Intelligence*, vol. 78, pp. 431–459, 1995.

[109] A. Peacock, D. Renshaw, and J. Hannah, "A fuzzy data fusion approach for image processing," *IEE Electronics Letters*, vol. 35, Sept. 1999.

[110] R. Jain and T. Binford, "Ignorance, myopia, and naivete in computer vision systems," *Computer Vision, Graphics, and Image Processing: Image Understanding*, vol. 53, pp. 112–117, Jan 1991.

[111] R. M. Haralick, "Computer vision theory: the lack thereof," in *Proc. Third Workshop on Computer Vision: Representation and Control*, pp. 113–121, IEEE Copumter Society, October 1985.

[112] K. Price, "I've seen your demo: so what?," in *Proc. Third Workshop on Computer Vision: Representation and control*, pp. 122–124, IEEE Computer Society, 1985.

[113] W. Foerstner, "10 pros and cons against performance characterization of vision algorithms," in *Proc. Workshop on Performance Characteristics of Vision Algroithms*, ECCV, Cambridge, April 1996.

[114] I. M. R. Labs, "http://intel.com/research/mrl/research/opencv/," 2000.

[115] M. Robinson, "http://www.ee.ed.ac.uk/~mer/project/project1.html," 2001.

[116] S. P. E. Corporation, "http://www.specbench.org," 2001.

[117] C. S. Regazzoni, C. Sacchi, and G. Gera, "Intelligence distribution of a third generation people counting system transmitting information over an urban digital radio link," in

Proc. 2nd European Workshop on Advanced Video-Based surveillance Systems, (Kingston), pp. 53 – 67, Kluwer, Sept. 2001.

[118] F. Oberti, G. Ferrari, and C. S. Regazzoni, "A comparison between continuous and burst, recognition driven transmission policies in distributed third generation surveillance systems," in *Proc. 2nd European Workshop on Advanced Video-Based surveillance Systems*, (Kingston), pp. 69 – 80, Kluwer, Sept. 2001.