Real-time, image-based Motion Estimation for the Dervish landmine-clearance vehicle

Christopher D Haworth

A thesis submitted for the degree of Doctor of Philosophy.
The University of Edinburgh.
September 2002

Abstract

Image sequence Motion Estimation has been an active and important area of research since the late 1970's. Areas for application include video coding, satellite remote sensing, CCTV surveillance, vehicle navigation and robot guidance. The motion estimation techniques for each of these areas share a common base but are often specialised for the particular problem. Furthermore, the computational requirements are often considered second behind the reported accuracy. However, for a real world application it is important to consider all aspects of the motion estimation technique together.

This thesis proposes and demonstrates that a low-cost commercial DSP platform can be used to perform real-time motion estimation for the Dervish landmine-clearance vehicle based on digital imaging of the ground. The Dervish landmine-clearance vehicle provides specific *application* requirements that are first examined in detail. The Motion Estimation *algorithm* requirements are then examined and, based on the tracking accuracy and theoretical computational performance, correlation-based feature tracking is selected. The individual parameters, including block size, motion model and interpolation technique, are carefully examined and suitable values selected. Finally, the *hardware* requirements are examined, a DSP platform chosen and a trial implementation reported.

A difficult problem within correlation-based motion estimation is how to update the reference block. This thesis describes novel methods for reference block updating. The reference block update strategies are examined in detail for parameter selection, tracking accuracy and computational performance. Particular attention is given to the tracking accuracy under increasing levels of image noise.

Using the *application*, *algorithm* and *hardware* requirements and decisions, a system is proposed that meets the main requirements of the Dervish landmine-clearance vehicle. The issues of robustness within long-term tracking and a novel dissimilarity measure based on the reference block update strategies are also presented. Parts of the research presented in this thesis have also been reported in conference and journal papers, which are listed at the end of the document.

Declaration of originality

I hereby declare that the research recorded in this thesis and the thesis itself was composed and originated entirely by myself in the Department of Electronics and Electrical Engineering at The University of Edinburgh.

Christopher D Haworth

Acknowledgements

Firstly, I would like to thank my two supervisors Dr David Renshaw and Dr John Hannah for their invaluable advice, guidance and support throughout the last 3 years.

I would like to thank Professor Stephen Salter for introducing me to the Dervish landmine-clearance vehicle and the idea of a vision-based short-range navigation system. I would also like to thank Dr Jimmy Dripps and Carn Gibson for their help in capturing video footage from their field-trials.

Finally, I would like to thank Dr Andrew Peacock, Peter Hillman and Mark Sinclair for the endless hours of discussion they provided on various topics throughout this thesis.

My thesis has been supported by EPSRC under studentship award number 99303062.

Contents

List of figures

List of tables

List of algorithms

Acronyms and abbreviations

ALU	Algorithmic Logic Unit
ASIC	Application Specific Integrated Circuit
BITSI	Proprietary SHARC module connection
BMA	Block Matching Algorithm
CCD	Charge Coupled Device, a form of digital imaging device
CIF	Common Interchange Format, a video format for digital cameras
CMOS	Complementary Metal Oxide Semiconductor, used in a form of digital imaging device
CMYK	Cyan Magenta Yellow blacK, a colour model normally used for printing
cPCI	compact Peripheral Component Interface
DCT	Discrete Cosine Transform
DLS	Discounted Least Squares, a recursive polynomial filter
DM	Data Memory bank of the Analog Devices SHARC
DRAM	Dynamic Random Access Memory
DSP	Digital Signal Processor
EMP	Expanding Memory Polynomial, a recursive polynomial filter
FIR	Finite Impulse Response, a non-recursive filter form
FPGA	Field Programmable Gate Array
FSA	Full Search Algorithm
HSV	Hue Saturation Value, an alternative colour model
IIR	Infinite Impulse Response, a recursive filter form
KLT	Kanade-Lukas-Tomasi feature tracking algorithm
LS	Least-Squares, used in a simple form of regression
MAC	Multiply-Accumulate
MAD	Mean Absolute Difference
ML	Maximum Likelihood, used in a more complex form of regression
MPEG	Motion Picture Expert Group, a common video compression standard
NTSS	New Three Step Search
PCB	Printed Circuit Board
PCI	Peripheral Component Interface

PDC	Pel (pixel) Difference Classification
PDF	Probability Density Function
PM	Program Memory bank of the Analog Devices SHARC
PMC	PCI Mezzanine Card
PSNR	Peak Signal to Noise Ratio
RAM	Random Access Memory
RGB	Red Green Blue, a common colour model
SAD	Sum of Absolute Differences
SDRAM	Synchronous Dynamic Random Access Memory
SHARC	Super Harvard ARChitecture
SHARCPAC	Proprietary SHARC module connection
SIMD	Single Instruction Multiple Data
SISD	Single Instruction Single Data
SRAM	Static Random Access Memory
SSD	Sum of Squared Differences
TSS	Three Step Search
UESA	Unimodal Error Surface Assumption
USB	Universal Serial Bus
VLIW	Very Long Instruction Word
VLSI	Very Large Scale Integration
VRF	Variance Reduction Factor
ZNCC	Zero-mean Normalised Correlation Coefficient

Nomenclature

β	Filter coefficient in the optimized FIR filter
θ	Recursive filter coefficient
κ_{scale}	Scaling coefficient within the rigid-body transformation
σ_n	Standard deviation of image noise
$\sigma_{n+1,n}$	Standard deviation of one-step filter prediction error
A	Computational cost of addition within a reference block filter
C	Computational cost of the correlation similarity measure
d	FIR filter depth
$\mathbf{d}$	Motion vector with components (u, v)
E	Computational cost of extraction of a rotated region
$\mathcal{I}$	Instantaneous image
$\mathcal{I}(\mathbf{x}, t)$	Pixel intensity from image $\mathcal{I}$ at position $\mathbf{x}$ and time t
M	Computational cost of multiplication within a reference block filter
M_t	Block from the position of best match in the image at time t
n	Translation search radius
n_m	Translation search radius at level m of the hierarchical pyramid
N	Block Width
r	Rotation search radius
r_m	Rotation search radius at level m of the hierarchical pyramid
$\mathbf{R}$	Rotation transformation matrix
$\mathcal{R}$	Reference block for use in correlation-based matching
R_t	Reference block prediction from a one-step filter at time t
T	Sampling rate for filter input
$\mathbf{T}$	Translation transformation matrix
Δt	Time increment between frames
$\mathbf{x}$	Position vector with components (x, y)

Chapter 1
Introduction

1.1 Introduction

Motion Estimation has been an active, interesting and important area of research since the 1970's and has many applications including surveillance, tracking, video compression and navigation. Autonomous mobile robots provide a fascinating but incredibly challenging application that has generated interest for decades. The movement seen by a vehicle, or robot, mounted camera that is due to the physical movement of the camera is referred to in this thesis as ego-motion.

This thesis investigates the application of motion estimation to a practical vehicle currently under development in the Department of Mechanical Engineering at the University of Edinburgh. The main aim of this vehicle is to aid in the clearance of landmines, dramatically increasing the speed of operation while reducing the risk to human operators.

1.2 Dervish landmine-clearance vehicle

There are many situations where an autonomous vehicle or robot may be preferable to a human. There has been recent research into developing anti-personnel landmine detection methods, such as ground penetrating radar. The major problem with these systems is that they must be placed relatively close to the ground but without risk of damage. Currently mini-zeppelins have been employed in this role and perform the task with some success. However, anti-personnel landmines present a serious threat to human landmine clearance teams and the Dervish landmine-clearance vehicle [1–3] is designed to reduce the risk to humans by detonating mines. It is possible that it could also be used as a platform to carry ground penetrating radar to identify the location of any mines that it could not detonate.

The Dervish landmine-clearance vehicle is a tri-wheeled structure, which is shown in Figure 1.1. Its motion roughly describes a series of circles, progressing forward, similar to a whirling

dervish. The narrow steel wheels, seen in Figure 1.2, both trigger the explosion and withstand the blast. The open-frame design of the vehicle itself allows the vast majority of the blast to pass through with only very minor structural damage. However, to ensure reliable coverage of the ground it is important that the tracks are placed at regular intervals, no greater than the smallest landmine trigger plate apart. The forward motion is controlled by altering the power in each wheel, in effect causing the Dervish to *skid* forward.

The mechanical design of Dervish will not allow it to operate in excessively rough terrain. However, after a conflict one of the most pressing problems can be feeding the local population, especially since prime agricultural land is often mined. Therefore, the Dervish landmine-clearance vehicle is designed to help speed up the clearance of anti-personnel landmines from flat arable land.

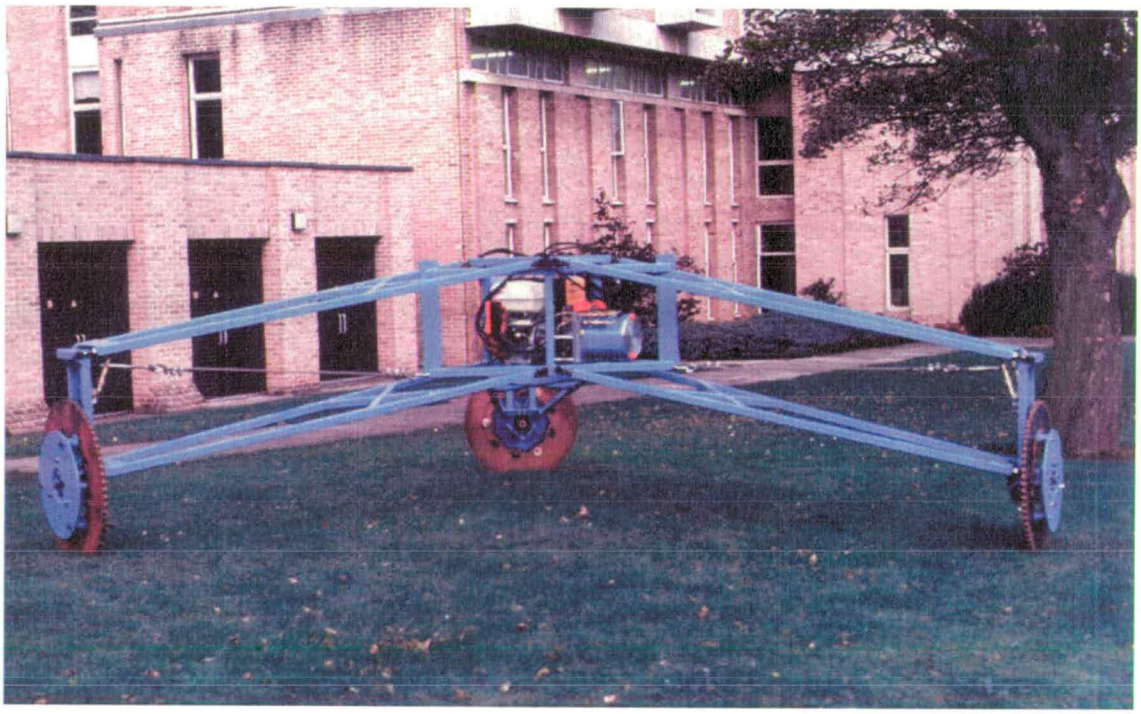


Figure 1.1: *A picture of the Dervish Anti-Personnel landmine-clearance vehicle Mk III. This remotely controlled vehicle has a unique tri-wheeled design with a complex motion pattern.*

The ground coverage from Dervish is naturally very good when travelling in a straight line due to the nature of its movement pattern every area is covered twice, by both the leading and trailing edge of the vehicle. However, the complex system of altering power in each wheel makes it very difficult to control. The main navigation system is to be a DECCA-based long-range navigation system that is, in theory, capable of millimetre accuracy. Early field-trials

Figure 1.2: *The closely space tracks that the Dervish landmine-clearance vehicle places can be clearly seen in soft ground. It is vital that the tracks are placed close together, otherwise a mine may be missed.*

have gone well but a second totally independent system has been proposed that could give highly-accurate short-range navigation information. One method for doing this would be a vision-based motion estimation system using a downward looking camera placed at the centre of the structure.

1.3 Image Processing Motivation

Motion Estimation has been an active research field since the late 1970's and many different algorithms have been proposed. These algorithms include correlation, differential and feature-based approaches for which Beauchemin and Barron [4] presented a recent survey; these techniques are discussed in more detail in Chapter 3. However, as noted by Touretzky *et al.* [5] in 1994:

> "...the real problem with video cameras is that image processing is computationally expensive. Even something as simple as calculating real-time optic flow requires more processing power than is practical for a mobile robot."

More recently, with increasing computational power, research has started to focus on practical real-time approaches rather than applications where expensive custom-design hardware is required. A vision-based short-range navigation system for the Dervish landmine-clearance

vehicle presents one such challenge where the constraints of the system mean that most current solutions are either too expensive or inappropriate.

The Dervish landmine-clearance vehicle will, necessarily, be operating in highly hazardous situations. Every possible consideration has gone into minimizing danger from the mine detonation but it is not possible to completely eliminate damage. Furthermore, many other problems exist when operating in an area of conflict. For these reasons, the design of the Dervish aims to be as low-cost as possible using easily replaceable off-the-shelf components.

The other major consideration is that operation will have to be accurate over extended periods of time and perform in real-time on the chosen hardware platform. Therefore, the three main constraints on the vision system imposed by Dervish are:

- Low-cost off-the-shelf embedded PC design
- Real-time operation on the chosen hardware platform
- Accurate operation over an extended period

1.4 Contributions

The main contribution of this thesis is to present a novel adaptation of correlation-based motion estimation techniques that allows accurate and efficient operation on the Dervish landmine-clearance vehicle. The competing requirements from the Dervish *application*, image processing *algorithm* and DSP *hardware* are examined in detail and a system proposed. Using the DSP hardware, which was originally identified in 1999, this thesis demonstrates that a low-cost commercial DSP platform can be used to perform real-time motion estimation for the Dervish landmine-clearance vehicle based on digital imaging of the ground.

Furthermore, extensive testing has been performed on the Dervish field trial data-sets that demonstrates the merits of the proposed system under a variety of conditions.

A difficult problem within correlation-based motion estimation is how to update the reference block. This thesis describes novel methods for reference block updating. The reference block updating strategies are examined in detail for parameter selection, tracking accuracy and computational performance. Particular attention is given to the tracking accuracy under increasing levels of image noise. A novel reference block updating strategy is also proposed for use in the

motion estimation, which allows systematic selection of parameters and correct track-initiation while achieving performance comparable to previous strategies.

1.5 Structure

There are three distinct sets of competing requirements for this project: *application*, *algorithm* and *hardware*. The decision making process is complex with factors from each of the three areas effecting the final outcome. To provide clarity this thesis presents the three areas in separate chapters. The structure of the thesis is as follows.

Chapter 2 starts with a review of the Dervish landmine-clearance vehicle *application* and the issues surrounding its development. An important stage for the entire project was collection and processing of Dervish field-trial data sets. The process is detailed in this chapter with further details given in Appendix C.

Chapter 3 gives an overview of Motion Estimation *algorithms* for vehicle navigation. This is broken down into two stages of, firstly, obtaining the image-based motion estimates and, secondly, translating these into the real-world ego-motion of the vehicle.

Chapter 4 extends work on reference block updating for correlation-based motion estimation. A new least-squares polynomial filter is proposed that allows systematic selection of filter parameters. Furthermore, a suitable track initiation system is described for the filter. The filter is applicable to all correlation-based motion estimation or tracking systems.

Chapter 5 provides a review of possible *hardware* platforms and after careful consideration an appropriate hardware platform is chosen. This DSP platform is introduced and basic details given, with Appendix D providing detailed information on the hardware platform. A number of specific design issues are also presented in this chapter.

Chapter 6 starts with an analysis of the Dervish landmine-clearance motion pattern. Taking the current correlation-based motion estimation algorithms, a novel extension is proposed that allows efficient frame-to-frame tracking for the application. The overall cost of the frame-to-frame tracking is then broken down and examined on the specific hardware platform. A trade-off between accuracy and efficiency is made and parameters are selected. Results for short-term tracking of Dervish field-trial data sets are presented.

Chapter 7 takes the frame-to-frame correlation-based motion estimation algorithm and extends this into a complete system for Dervish vehicle navigation. Results are presented on the accuracy of the proposed algorithm in comparison to other relevant techniques. The efficiency of the proposed algorithm on the DSP platform is then examined and some conclusions reached on the current choice of hardware.

Chapter 8 provides a conclusion and summary of the thesis, along with areas that remain for future research based on the outcome of this work.

1.6 Summary

The subject.of this thesis is motion estimation applied to the Dervish landmine-clearance vehicle. This is motivated by the need for a low-cost real-time short-range navigation system. In contrast current motion estimation techniques have often concentrated on more theoretical challenges instead of a practical solution.

The thesis makes contributions to the Motion Estimation field both with some novel adaptations to the current algorithms and a novel implementation of the proposed algorithm on a practical, low-cost hardware platform. The reference block updating, presented in Chapter 4, provides an important contribution that is applicable to the wider field of region-based motion estimation.

Chapter 2
Dervish land-mine-clearance vehicle

2.1 Introduction

This chapter introduces the Dervish landmine-clearance vehicle in more detail and discusses some of the engineering tasks surrounding the development. The constraints imposed by Dervish and their effect on any vision-based navigation system are outlined in Section 2.2. The complex choice of hardware platform and image processing algorithm is also introduced in Section 2.2. Chapter 3 presents a review of motion estimation algorithms for vehicle navigation and Chapter 5 presents a review of possible hardware platforms. From the information in these two chapters a suitable motion estimation algorithm and hardware platform are chosen. Section 2.3 describes the Dervish field-trials that have taken place and what data has been gained from these. It details how this data is used in testing for the accuracy of the motion estimation algorithms.

2.2 Dervish Overview

The Dervish landmine-clearance vehicle is designed to sweep areas of suspected land and detonate any anti-personnel landmines. Due to the very large array of mine technology it is very difficult to reliably detect anti-personnel landmines, although recently some very promising results have been reported using a mini-zeppelin mounted ground-penetrating radar. Dervish instead sweeps the entire ground, detonating all mines that it runs over. It is important that the tracks are placed at regular intervals, no greater apart than the smallest landmine trigger plate, otherwise ground coverage cannot be guaranteed. Figure 1.2 shows the wheel tracks laid down in soft-ground during field-trials. There are many complex design challenges associated with the Dervish vehicle but a summary of the key technical points is presented below. One key point is that the Dervish must be very low-cost, with the total cost in thousands of pounds, rather than the hundreds of thousands of pounds associated with current mine clearance operations. If the concept was proven technically successful and the cost was low enough to allow post-conflict

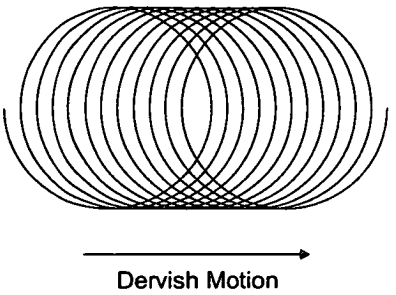


Figure 2.1: *The circular motion that the Dervish wheels undergoes during normal operation is shown. The spacing between tracks has been increased for clarity but in practice it is important that there are no gaps in the wheel tracks laid down.*

countries to purchase then the Dervish could prove a very valuable resource to demining teams. The vehicle and all its systems are currently under development and it is possible that these details will change as a result of future work.

2.2.1 The Mechanics of Dervish

The Dervish landmine-clearance vehicle is a tri-wheeled structure (shown in Figure 1.1). There is no steering but instead each wheel power is independently controlled and by varying the power of the wheels through each third of the rotation it is possible to *push* the vehicle in one direction, where the motion roughly describes a series of circles. The motion of the wheels is shown in Figure 2.1; the spacing between tracks has been increased for clarity but in practice it is important that there are no gaps in the tracks laid down. The power to the three wheels is controlled by an offset sine-wave input, as shown in Figure 2.2. One inherent problem of this control system is that a dynamic control model for the motion is very difficult to construct. Combined with the difficult operating terrain it is necessary to use a feed-back control system to achieve the required accuracy.

The majority of anti-personnel landmines concentrate a small proportion of their blast upwards and the remainder of the force propels shrapnel outwards. The dimensions and open-structure of Dervish are such that the vast majority of the shrapnel will pass clean through the open structure causing practically no damage. However, to reduce the effect of the upwards blast the Dervish must have very thin and very tough steel wheels attached with minimum complexity to the thin, almost scaffold like, structure. Any attempt to add more complex wheel design, in particular steering or suspension, simply provides a much greater area that is susceptible to damage.

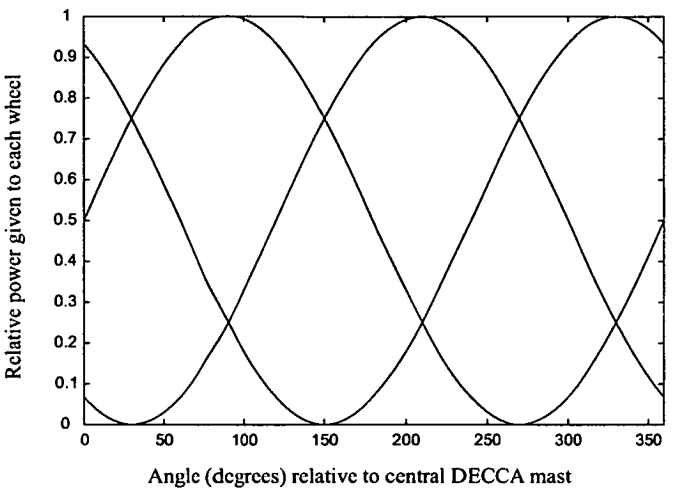


Figure 2.2: *The power given to each wheel (red, blue, green) is shown as the vehicle rotates, relative to a central mast. The power follows a sine-wave pattern, where the total power remains the same but the power in a given direction can be varied.*

Anti-tank landmines are easier to detect and are laid in much smaller quantities than anti-personnel landmines. Furthermore, they will not detonate unless a large force is applied; for example greater than $150kg$. Therefore, they are much less of a problem in landmine clearance. However, their massive blast is concentrated in a very small area upwards, making them very dangerous to any vehicle that runs over them. For this reason, it was decided that Dervish would attempt to trigger anti-personnel landmines but not anti-tank landmines. This requires that Dervish exerts a force no greater than $100kg$ per wheel.

One of the first stages of development was field-trial testing with anti-personnel landmines. This demonstrated that the Dervish could withstand detonation of a landmine and then continue operating undamaged [1]. The only structural damage sustained was about 1mm of steel taken off the wheel surface, but with $200mm$ of solid steel on the edge of each wheel the Dervish could detonate many landmines before requiring replacement wheels. When Dervish was tested with an anti-tank landmine the whole structure was blown apart and no trace of the individual steel wheel that triggered detonation was ever found, confirming the decision to avoid detonating anti-tank landmines.

A problem that was encountered in early field-trials was a slight *circular wobble* within the motion of the Dervish, which is shown in Figure 2.3. The effect of the *circular wobble* perpendicular to direction of travel is to move the centre of the vehicle from side to side, approximately 1m total. This has very little effect on the operation of Dervish. However, the effect of the *cir-*

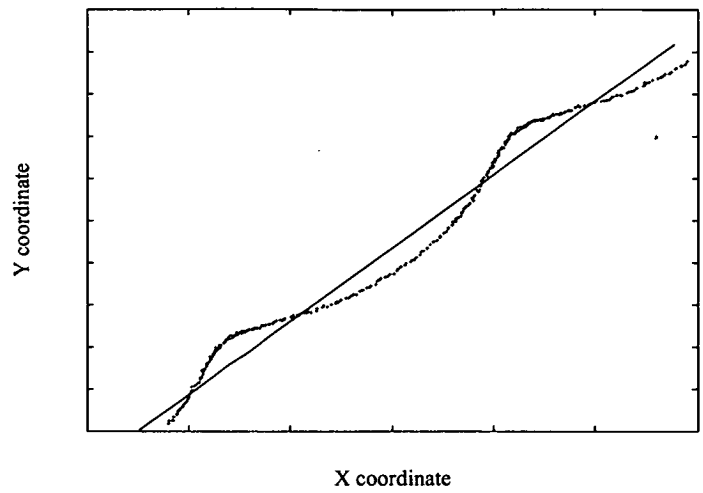


Figure 2.3: *A simulation of the* circular wobble *seen in Dervish field-trials. The green line shows the programmed path while the red crosses show the actual position every* 25^{th} *of second. The varying density in the direction of travel shows the wobble affecting forward motion as well as introducing sideways motion.*

cular wobble on the motion in the direction of travel is on the travelling speed, which slows down and speeds up. Therefore, if adequate ground coverage is to be achieved it is very important that the maximum speed, including extraneous motion, does not extend the distance between tracks beyond the minimum trigger plate size.

Two possible causes can be identified for the *circular wobble*: wheel angle and power input. After some experimentation in field-trials it was discovered that the design is very susceptible to small changes in the angle of the wheels. All three wheels should be exactly perpendicular to the main structure but small differences occur in practice. Furthermore, the theoretical sine-wave power input shown in Figure 2.2 proved difficult to use in practice because the hydraulic pumps used to drive the wheels do not have good linear operation at the bottom end of their range. To avoid this problem the sine-wave power input was chosen to operate in the range $[2.5, 5]$ volts rather than $[0, 5]$ volts. This leaves some residual power when, in theory, the power should be turned off. It is expected that alternative hydraulic pumps will be employed, when the necessary design is complete. However, at this stage it is necessary to assume that the motion seen in field-trials, including the *circular wobble*, will be the final motion pattern.

2.2.2 Dervish Navigation and Mapping System

The engine and control system for Dervish are housed within an armoured central box that is fairly restrictive in space. The main control system will be based around an embedded PC and all other systems are expected to connect directly to this. A development version of this control system had already been designed at the start of this project.

The central navigation system will have a pre-defined route plan downloaded that provides good ground coverage of an area to be sweeped but avoids any obstacles that the Dervish could not traverse. The original long-range navigation system is based on a variant of the DECCA navigation system that is, in theory, capable of millimetre accuracy; this is being developed within the Department of Electronics and Electrical Engineering [6]. It uses patterns of orthogonal microwaves transmitted from 3 control towers placed around the mine-field to setup a standing pattern of microwaves, generating a mesh. The on-board Dervish receiver is programmed to lock onto a node of the mesh. The mesh is then slowly shifted by altering the transmitted signals so that the Dervish moves along the ground, following the node. In practice the accuracy has been difficult to establish and jumping from one node to another has been observed; there is no way of detecting this from within the DECCA system. It has been proposed that a second short-range navigation system could provide high-accuracy estimates over a limited period of time to provide confirmation of the true motion for the Dervish vehicle. The two navigation systems would provide results to the central navigation system, which provides a final estimate for the ego-motion of the Dervish. This information is passed onto a mapping system to determine what ground coverage can be guaranteed from the current Dervish sweep. This is set-up is shown in Figure 2.4, with the external telemetry output shown for each stage.

A critical question for the navigation system, given the speed of development in alternative technologies, is whether the DECCA-based system is still the best alternative. In particular, commercial GPS system can now report accuracy down to the nearest 10cm. If this was combined with a short-range navigation system, such as the proposed vision-based navigation system, then it may prove more effective and it would certainly simplify some of the development issues. Another area that was not considered in the original design of the Dervish is the inclusion of on-board inertial navigation sensors. There is a large area of research focusing on the fusion of multiple sensors and significant improvements in overall accuracy have been reported [7]. The inclusion of a dynamic control model would also increase the level of accuracy and a preliminary investigation into this was undertaken. However, as mentioned in Section 2.2.1,

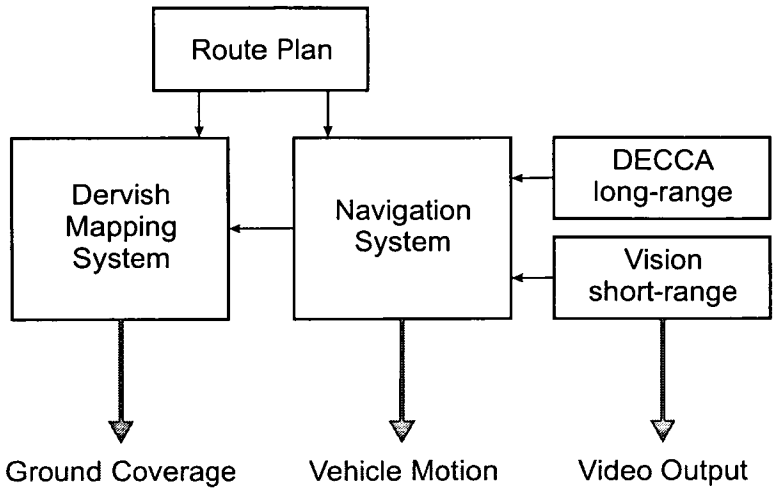


Figure 2.4: *An overview of the complete Dervish navigation and mapping systems is shown. The two navigation systems can be seen along with the route plan and mapping system.*

constructing such a model is very difficult due to the unusual mechanical construction. This is further complicated by large level of unpredictable motion seen in Dervish field-trials, mainly due to the necessity of operating with thin steel wheels in difficult operating terrain. For these reasons, the use of a dynamic control model is not considered further in this thesis. Should the current system be re-examined then a multiple sensor arrangement incorporating GPS, Vision and inertial sensors will probably provide the highest accuracy solution at an acceptable cost.

2.2.3 Vision-based Navigation System

A vision-based, short-range navigation system was proposed to augment the DECCA-based long-range navigation system. The camera is mounted at the centre of the vehicle looking downwards and encompassing the centre of rotation. Figure 2.5 shows a simplified diagram of the Dervish with the proposed camera setup and Figure 2.6 shows a simulated image sequence. The camera will have to be low-cost, due to the continual danger of damage, and the motion estimation must be performed on-board. From the original briefing it appears that Dervish landmine-clearance project imposes the following constraints on the system (originally given in Chapter 1):

- Low-cost off-the-shelf embedded PC design
- Real-time operation on the chosen hardware platform

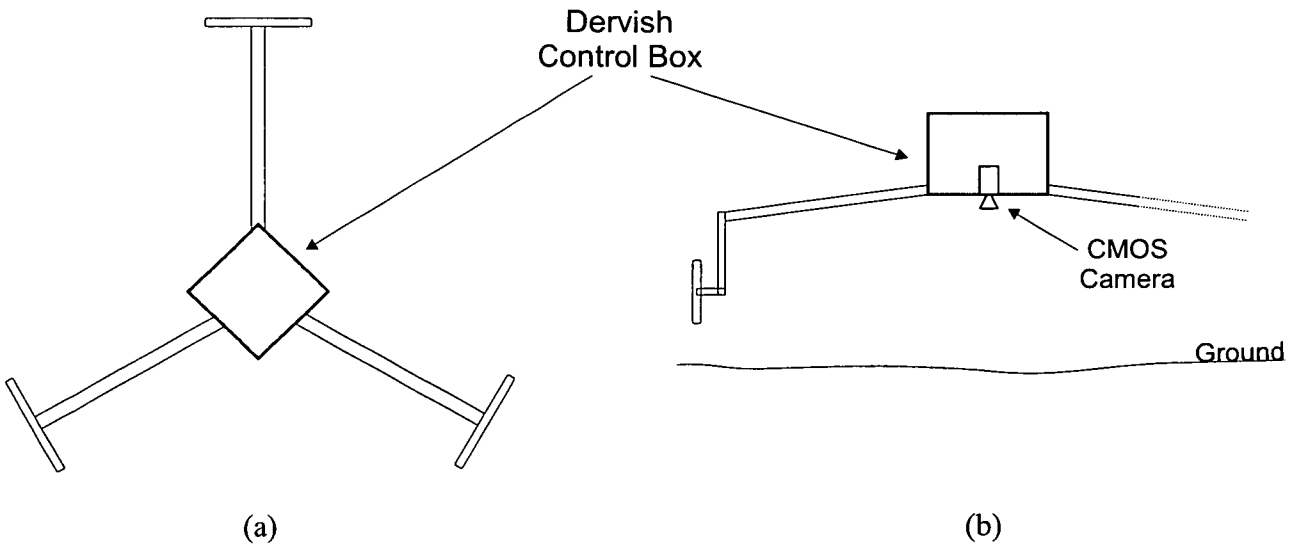


Figure 2.5: *The proposed camera set-up for the Dervish landmine-clearance vehicle. The downwards facing camera is placed at the centre of the structure away from any mine explosions.*

- Accurate operation over an extended period

The first of these three constraints is very important because there is limited space within the Dervish and the other control systems have been designed around an embedded PC. It is important that the vision-based navigation system can connect directly to the main control systems using either a PCI or, possibly, compact PCI (cPCI) connector. The overall cost of a Dervish vehicle aims to be under 5000 pounds since the vehicle will be in danger of damage on a daily basis. The exact budget available for image-processing hardware has not been determined but the overall cost should be kept as low as possible. Originally it was envisaged that the Dervish would only be produced in low volumes at irregular intervals with a maximum of twenty Dervish vehicles predicted. In this volume it would not be possible to minimize extra hardware cost through higher volume production or purchasing. Given the extent of the landmine problem, which is still increasing, it is possible that larger numbers of Dervish may be produced should the concept be proven technically successful. However, due to the highly specialist nature of landmine clearance operations it is unlikely that the Dervish would ever be produced in large enough quantities to all any form of large scale production. Therefore, to meet the first constraint it is important to utilise low-cost, off-the-shelf components that help to minimize the engineering complexity. Essentially a *plug and play* solution where very limited custom hardware and interfacing is required would provide the ideal solution.

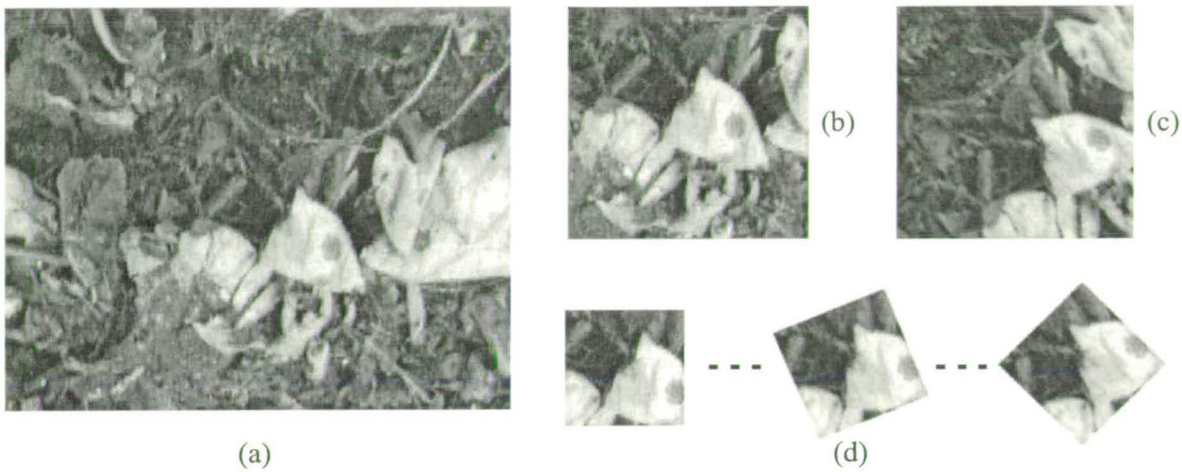


Figure 2.6: *Simulated image sequence (a), showing the reference image (b) and the current image (c). Three different rotated blocks that have been generated from the reference image are shown (d).*

The last two constraints on the system are that it achieves real-time performance with the required degree of accuracy for an extended period. It is at this stage that the choice of motion estimation algorithm and hardware platform become linked in a very complex manner. In Chapter 3 many different motion estimation algorithms are reviewed and all are mature research fields. However, Chapter 3 goes on to select a set of techniques most suitable for the Dervish application.

Achieving the desired accuracy at real-time performance is the main focus of this thesis and the discussion and results are presented throughout the remaining chapters. Therefore, the system accuracy and performance are not considered further at this stage. However, real-time operation for the Dervish is defined and the requirements of the motion estimation algorithm on the hardware platform are presented below.

Real-time Performance

The definition of real-time performance requires some discussion and justification. From the landmine clearance perspective the system must be able to guarantee the ground it has covered and, therefore, must be receiving updates regularly enough to confirm that the spacing between wheels tracks is less than the smallest landmine trigger plate. With the smallest trigger plate size of $10mm$ and a wheel width of $50mm$, the maximum distance between the centre of wheel tracks is $30mm$ [1]. However, in practice, Dervish has a fairly slow movement pattern

with a rotational speed of approximately 0.5 radians per second and a translational speed of approximately $100mm$ per second. Therefore, the maximum motion between frames at 5fps is $20mm$ and 0.1 radians, which is small enough to allow the required motion to be confirmed. An upper limit is set by the frame rate for digital video cameras, which is between 15fps and 30fps depending on the model chosen.

While in theory a higher frame-rate is always desirable, it is important to allow enough time for significant ($>$1-pixel) movement between frames to occur because the noise level in the video sequence is very high and detection of sub-pixel motion would be difficult. Any increase in frame-rate also requires an increase in I/O throughput that would have to be accounted for. Furthermore, if the system performance exceeded requirements then a lower performance hardware platform could be utilised to reduce the overall cost; for example, a 4 DSP platform that performed at 24 frames per second could be reduced to a 2 DSP platform with a notional performance of 12 frames per second.

Real-time performance is commonly divided into *soft* and *hard* real-time [8], with the distinction that in hard real-time performance the system must respond to a stimulus within a fixed-amount of time to avoid catastrophic system failure. In a soft real-time system, failure to respond will produce a degradation in system performance instead. The Dervish short-range navigation system fits into the soft real-time category, where failure of a individual navigation system does not compromise system safety. However, very high levels of accuracy are required in the navigation systems and use of best-guess data, even for very short periods, is not acceptable. Any degradation in navigation system performance, even a delay in delivery of the motion estimate, will result in the navigation system terminating the current mine clearance sweep. The Dervish vehicle would then need to return to a known position and restart the mine clearance sweep. The long-range navigation system will always be able to return the vehicle to base even if the short-range system fails.

Therefore, the real-time performance requires that a frame-rate of between 5 and 20 fps is achieved. The remainder of the Dervish system must also be aware of the frame rate, which must be consistent. Finally, it is important that the short-range navigation system reliably detects any failures to ensure correct operation of all other systems.

Image Processing Constraints

The real-time performance detailed above is a fundamental requirement so the motion estimation algorithm and hardware platform must be chosen together because the motion estimation algorithms all differ in their basic processing requirement. Some algorithms consist of simple calculations and are inherently parallel, making them highly appropriate for implementation in custom hardware. Other algorithms have far more complex calculations and branching structure for which a high-speed sequential processor may prove advantageous. Any decision on either hardware platform is entirely dependent on the motion estimation algorithm and to a lesser extent vice versa. Chapter 3 provides a summary of motion estimation algorithms and concludes by selecting a set of appropriate techniques. Chapter 5 then goes on to select a suitable hardware platform for the Dervish application. Reviewing Figure 2.6, two things that can be noted at this stage are that the motion consists of considerable rotation and that there is very limited depth variation in the normal field-of-view. This is unusual in a vehicle navigation system where the majority of motion would normally be forward translation and a considerable range of depth would be anticipated.

2.3 Dervish Field-trials

2.3.1 Field-trial Setup

The Dervish landmine-clearance vehicle has undergone numerous field-trials in development. The first stage of field-trials was to demonstrate that the open-structure and narrow steel wheels could trigger and survive multiple anti-personnel landmine detonations. This stage was wholly successful and video footage can be obtained on the Dervish landmine-clearance project website [1]. The second stage of testing, using Dervish Mk III, was to develop sub-systems for control and navigation. The main focus of the field-trials was to allow the development of the mechanics and DECCA navigation system. However, it was possible to mount a downward looking digital video camera on the Dervish and capture extensive footage. Due to physical and mechanical constraints within the Dervish the camera was mounted horizontally, pointed at a mirror set at 45 degrees. Although this restricts the effective field-of-view, in practice this setup would allow the lens and camera to be better protected from detonation damage. Figure 2.7 shows the field trial camera setup. The only data that could not be captured in field-trials was of a Dervish running over a mine, as no further explosive testing took place during the

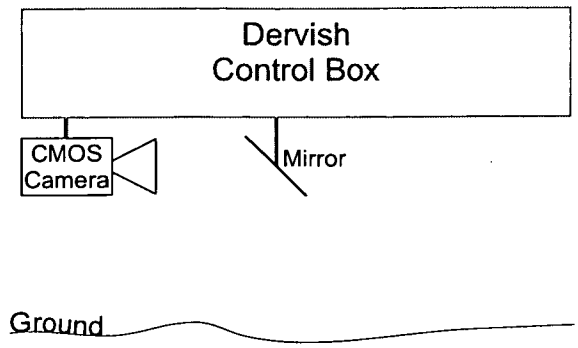


Figure 2.7: *The camera set-up used in the Dervish field-trials, showing the 45-degree mirror employed.*

course of the research reported in this thesis.

From the field-trial videos all movement rates and image data were as expected, except that a large amount of vibration can be seen. An attempt to counteract this was made by shock-mounting the main engine and damping the video-camera brackets but it was not possible to eliminate all vibration, in particular on tarmac or concrete surfaces where the skidding wheels naturally and unavoidably cause vibration.

2.3.2 Camera System

The digital video camera used was a 3-CCD Sony DCR-TRV890 with effective resolution of 720×576. It was necessary to maximise the field-of-view for the camera so the minimum focal length (widest angle) was used for all filming. The very high resolution, 3-CCD colour camera does not represent a realistic option for the final system, so the resolution was halved - also avoiding problems with the camera interlacing - and converted to greyscale using the standard *luma* colour conversion (Equation 2.1). Some noise reduction is noticeable with the resampling producing a low-pass filtering effect and it may be necessary to replicate this noise reduction for the final camera setup.

$$Y' = 0.299R + 0.587G + 0.114B \tag{2.1}$$

Therefore, the final format for the test data sets is an 8-bit greyscale image at a resolution of 360×288, which is half the original resolution. The final system is anticipated to work at

CIF resolution (352 × 288). A ruler was placed in the image, with markings at 5cm and 10cm distances. Using these measurements, it was possible to determine that the vertical resolution is 1.4mm per pixel and the horizontal resolution is 1.7mm per pixel, at the reduced resolution. Given a desired accuracy of ±10mm, the accuracy must be ±5.5-pixels horizontally and ±7-pixels vertically. For practical evaluation the mean squared error is use and this means that at no stage must the mean squared error exceed 9-pixels.

This camera setup is fairly high resolution given the small field-of-view and a wider field-of-view may be useful. However, it is not possible to widen the field-of-view further without using a shaped mirror under the Dervish. This was not possible within the field-trials and any warping introduced within the scene would have to be removed. The field-of-view is discussed further in Chapter 8.

2.3.3 CMOS Cameras

As stated in the previous section a 3-CCD camera was employed in field-trials providing a high-quality image with colour information at full spatial resolution. Section 3.2 provides a brief introduction to digital image capture devices and their respective advantages and disadvantages. It is clear that the camera employed in field-trials will not be employed within the final system. A single sensor CMOS camera represents the most likely option and this provides a major choice: colour (Bayer pattern) image sensor or grey-scale image-sensor. The problem with a Bayer pattern image sensor is that it reduces the actual spatial resolution of the image, to allow colour information to be captured. A considerable number of image artefacts are generated which must be removed through pre-processing. For these reasons, in particular the loss of spatial resolution, a grey-scale camera should be used, unless colour information is employed.

The CMOS camera will have to undergo camera calibration, similar to that undertaken for the field-trials. This could, for example, be performed every time the Dervish is reassembled in a new location. The exact nature of this camera calibration is not considered within this thesis but a non-technical automated procedure would be necessary. It will also be necessary to employ some form of range finder, such as a laser range finder, to determine that the distance to the ground has not changed during an operation. For example, Dervish may run over a large boulder which would significantly reduce the accuracy of any vision-based motion estimation. The exact affect of depth variation on the vision-based navigation accuracy will need to be investigated in the next stage of field-trials. However, there are no examples of this problem in

any of the current test data sets, which are described in the next section of this chapter.

2.3.4 Dervish Test Data Sets

Unfortunately, due to the focus of testing being on development of the mechanics and DECCA navigation system, only 5 days were available for capture of image sequences for this research. Although the field-trials produced a large number of different video sequences there were considerable problems in capturing them and they do not provide a complete set of possible situations. However, a representative sample, including different motion, terrain and lighting, has been selected for use in testing. Table 2.1 provides a summary of the Dervish test data set video sequences.

Sequence	Seconds	Rotation	Motion
Concrete	8	clockwise	slow sweep
LeavesStable	12	clockwise	sweep
LeavesUnstable	8	clockwise	sweep, rough ground
GrassRuler	10	anti–clockwise	sweep
Grass	7	anti–clockwise	sweep

Table 2.1: *Manually tracked data sets from the Dervish field-trials. The normal Dervish motion is a sweep of the ground but other movement can be seen.*

It proved difficult to obtain an accurate ground-truth for the Dervish field-trial video sequences. Various motion estimation techniques, including more complex image registration techniques, were tried but it was not possible to confirm that they were of any higher accuracy than the algorithms being tested. This was in a large part due to the vibration present in the sequences. Instead the sequences were manually tracked over a short period. Given an easily identifiable feature it is possible to guarantee that manual tracking will never accumulate more than a 1 or 2 pixel error. However, it is not possible to guarantee the accuracy of an individual frame, particularly under vibration. It was only possible to track sequences for limited periods of time, depending on the features within the scene and the number of frames tracked varies between sequences. Therefore, the manually tracked data can be used for comparison of techniques but care must be taken not to consider the individual frame accuracy as perfect. Appendix C presents more details on the manual tracking process.

The entire system requires a high level of accuracy over an extended period. Therefore, a set of simulated sequences was developed that aim to emulate the Dervish field-trial motion as accurately as possible over an extended period of operation. The simulation attempts to model the motion seen in field-trials and not the actual mechanics of Dervish operation. This allows testing of the accuracy of the feature tracking over sequences of a few minutes. Further technical details and full code for this simulator can be found in Appendix C.

The selection of a ground image for use in the Dervish simulator proved a difficult choice and it is important to remember that the Dervish simulator has been designed to reproduce image sequences with a similar motion pattern to the video sequences captured in the field-trials. No attempt has been made to accurately model the visual artefacts of the video sequences from the field-trials or the behaviour of Dervish and its subsystems. Use of a realistic ground view was considered but bearing in mind that the simulator is designed to test that the completed vision-based navigation system performs correctly over an extended period and that the accuracy of the feature tracking can be determined from the field-trial data sets. Therefore, the input to the ego-motion estimation from the feature tracking must model that seen in field-trials but the input to the feature tracking does not need to accurately model the real data. An attempt to accurately model the visual artefacts from the video sequence input into the feature tracking stage would require a very complex simulator and it would be very difficult to verify that the artificial image sequence were an accurate representation of the real data. Furthermore, capture of realistic terrain would require some form of mosaicing technique to piece together individual image sequences which could affect the accuracy. For these reasons, an aerial view of Edinburgh has been employed which does not contain any mosaicing artefacts.

2.4 Summary

This chapter has presented the main details of the Dervish landmine-clearance vehicle that are relevant to this project, which will be operating in a wide variety of situations, ranging from agricultural fields on a hill side (Kosovo) to dry, dusty flat-lands (Southern Afghanistan). The assumptions and constraints that can be made on the system have been detailed and a specification for correct operation identified. The maximum error in the estimation of motion between frames has been defined and an overall accuracy of ± 9-pixels for the mean squared error set as the target.

Finally, the chapter presented an overview of the Dervish field-trials which have taken place during the course of this project. The test data sets from the field-trials are detailed but long-term ground truth was not available for the field-trial data sets, so a simulator was designed to output data sequences, similar to the field-trial data. The simulated sequences, for which exact ground truth is known, can then be used to determine accuracy over an extended period of time. Any subsequent research would require access to the Dervish landmine-clearance vehicle for more extended testing. Chapter 3 goes on to review motion estimation algorithms and Chapter 5 reviews hardware requirements. Given the *application*, *algorithm* and *hardware* requirements, Chapter 6 and 7 detail more information about the vision-based short-range navigation system.

Chapter 3
Image Processing for Vehicle Navigation

3.1 Introduction

Motion detection, estimation and tracking are useful in a wide variety of applications including video surveillance, video coding, vehicle / robot navigation and satellite imagery. The taxonomy given in Figure 3.1 divides the research field into four key areas: detection, estimation, tracking and compensation. It is important to remember that these areas of research are closely related and are not always easy to distinguish from each other, particularly in real-world applications.

Motion detection is used to determine if motion is present between two video frames and the result is normally a region of interest indicating where motion occurred. A common example would be attempting to detect when changes in a scene are due to real-world motion and not, for example, shadows caused by changes in illumination. Motion estimation is the process of identifying motion between two frames within a region of interest. In this case the output is normally the magnitude and direction of the motion for the region of interest, either as a whole or on a per-pixel basis. Motion tracking attempts to track an object across multiple frames, involving further processing (identification, segmentation, filtering) of the motion estimates. The output from motion tracking would normally be the current position and velocity of the target object. Motion compensation, or motion compression, is used in video coding standards to reduce the size of the compressed sequence by removing as much temporal redundancy as possible.

As stated the techniques are used in a wide variety of applications, however, while they share common ground in the underlying algorithms employed, the final goal is not common. Often a technique from one area will be adapted to a totally new application; for example, pel-recursive video compensation forms the theoretical basis for differential optical flow techniques that are now prominent in areas such as robot navigation and structure-from-motion. However, it is

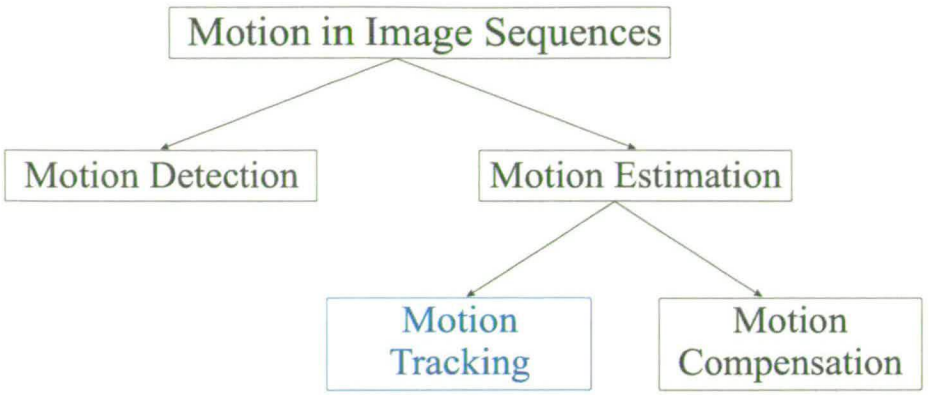


Figure 3.1: *A Motion Taxonomy for research in motion in image sequences, highlighting the position of Motion Tracking within the wider research area.*

necessary to remember that despite the common background the applications can have, either obviously or subtly, different aims. In particular the final measure of success between two seemingly similar techniques may be entirely different.

As well as the main task of motion estimation it is also necessary to first capture the digital image and afterwards process the individual motion estimates to produce an estimate of vehicle motion. The process of taking the individual motion estimates and producing a real-world estimate of ego-motion for the vehicle is referred to in this thesis as passive navigation. Figure 3.2 shows an overview of the vision-based vehicle navigation system. In this case the estimation of ego-motion is split into two independent phases: motion estimation and passive navigation. These are tackled separately with motion estimation addressed in Chapter 6 and passive navigation addressed in Chapter 7 of the thesis.

This chapter considers the problem of ego-motion estimation from a vehicle navigation perspective and uses the structure presented in Figure 3.2 as a taxonomy, first giving a brief look

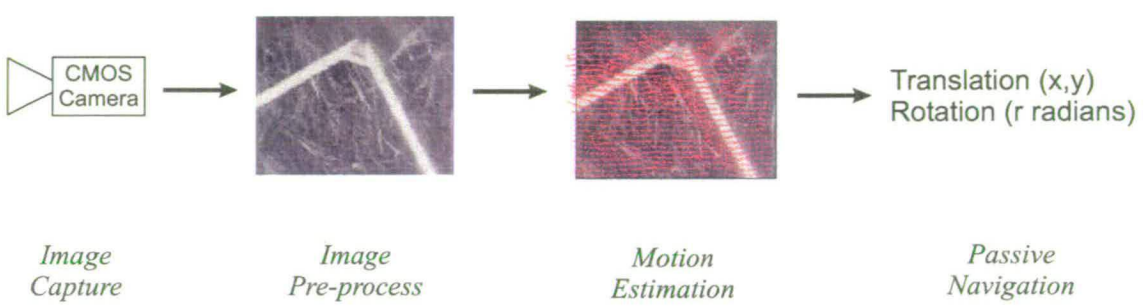


Figure 3.2: *An overview for vehicle navigation showing the image capture, pre-processing, motion estimation and passive navigation stages. An example of the output from each stage is shown.*

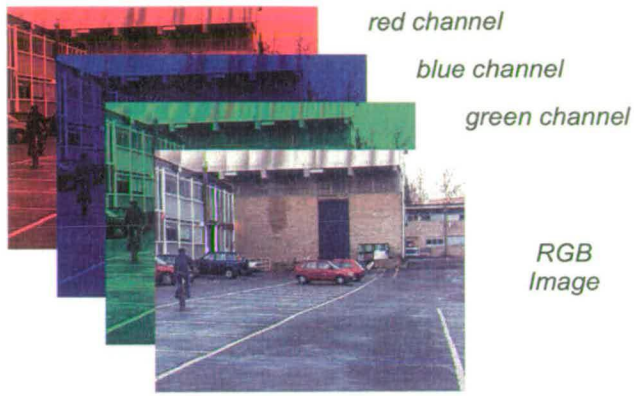


Figure 3.3: *A RGB colour image with the three separate colour channels shown.*

at digital image capture in Section 3.2. Next techniques for low-level motion estimation are examined in Section 3.3 before going on to examine the options for passive navigation in Section 3.4. Finally, Section 3.5 examines possible ego-motion estimation algorithms specifically for the Dervish landmine-clearance vehicle. Chapter 4 extends current reference block update strategies for correlation-based motion estimation before Chapter 5 examines possible hardware platforms for the Dervish landmine-clearance vehicle, including the suitability for use in image processing.

3.2 Digital Images

A greyscale digital image is normally represented as a two-dimensional array of intensity levels. A colour digital image is normally represented by three two-dimensional arrays of colour intensity levels; commonly red, green and blue. This representation is shown in Figure 3.3. There are other colour models, such as HSV and CMYK, with specific advantages and uses. However, before any image processing can be performed a necessary prerequisite is to capture and digitise the image.

An analog camera, such as a VHS Video Camera, could be used to capture the sequence, which must then be digitised using a frame-grabber. In this case the quality of both the camera and frame-grabber will effect the accuracy of the final output. There are many different analog video standards but a reasonable assumption is that the signal would be PAL / NTSC TV standard and have a roughly equivalent operating resolution. The characteristics of analog cameras, such as interlacing and motion blur, have been investigated carefully and are understood.

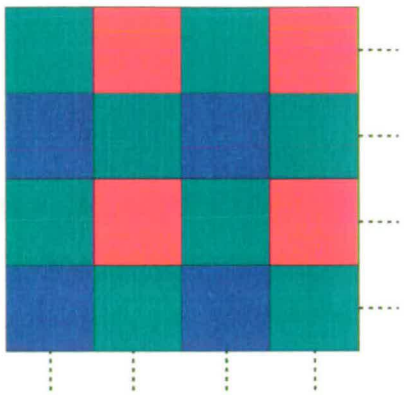


Figure 3.4: *A Bayer pattern digital image sensor is shown. This pattern allows a single image sensor to capture a reasonable quality colour image avoiding the size and cost of three separate image sensors.*

Alternatively, a digital camera, such as a MinvDV Digital Camera, could be used to capture the video sequence with the data being transferred to computer via USB or Firewire (IEEE1394). In this case the major consideration for accuracy would be the imaging device. A grey-scale sensor detects total light intensity at every location in the sensor array. To detect colour channels a filter must be applied, narrowing the bandwidth of intensity detected [9]. However, this requires three separate sensors to achieve full spatial resolution. Alternatively, a bayer-pattern can be employed, as shown in Figure 3.4 [10]. In this case, green is detected at $\frac{1}{2}$ spatial resolution and red and blue at $\frac{1}{4}$ of the full spatial resolution. The colour channels are then interpolated, however, a bayer-pattern sensor employs complex pre-processing designed to remove visual artefacts. Alternatives to the Bayer pattern exist, but are not yet common. Another issue with digital cameras is that a internal compression is normally performed to reduce storage requirements. Depending on the compression format it may significantly effect further image processing. For example, MPEG compression performs block-tracking, which will effect any further motion estimation. For all types of digital cameras, including three sensor and bayer-pattern cameras, it is important to remember that the output resolution is not the true silicon-level image resolution and that extensive pre-processing may have been performed, including compression.

3.3 Motion Estimation

The techniques for motion estimation can be divided into two distinct categories: optical flow techniques and feature correspondence techniques. In optical flow techniques the motion for

every point across the image is computed, giving a dense measurement of motion. In contrast, feature correspondence techniques identify key features in each image and match the corresponding features between consecutive images in the sequence. The identification of features is based around selecting features that can be unambiguously identified in subsequent frames.

During the literature review on motion estimation, it can be seen that feature correspondence and optical flow are useful in many of the same fields. One comparison previously made [11] is that optical flow requires closely spaced images while feature correspondence can work on larger spaced images. However, a more important difference is that optical-flow provides a far higher density of information, generally covering the entire image, by utilising certain constraints. Region-based feature correspondence attempts to identify areas of sufficiently high texture to be able to track accurately and independently through the sequence. The high-density of information in optical flow is not useful in this situation and complicates passive navigation. However, differential optical flow is an extremely important area within motion estimation and, for this reason, is reviewed in detail. Barron *et al.* [12] produced an in-depth survey of Differential, Correlation and Frequency based Optical Flow techniques in a technical report; this was later updated by Beauchemin and Barron [4]. Their taxonomy has been used for the optical flow section of this chapter.

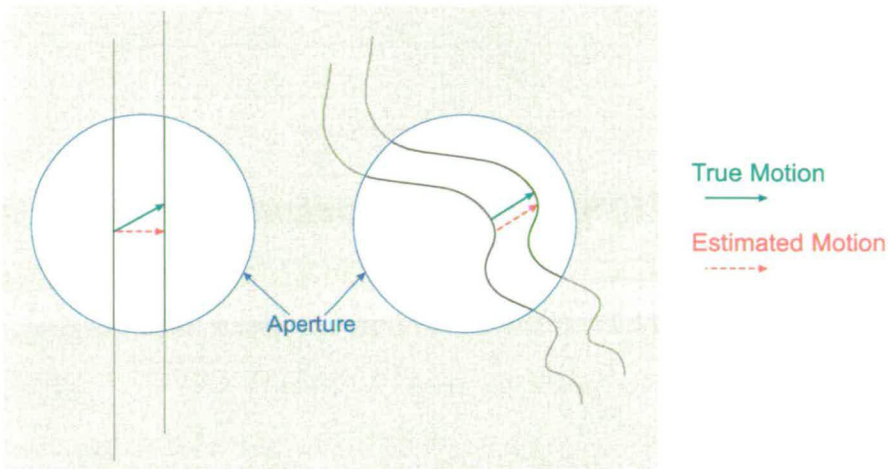
3.3.1 Differential-based Optical Flow

Differential-based optical flow techniques are similar to pel-recursive (pixel recursive) techniques developed in the late 1970's for video coding, which compute the motion from the element difference signal (spatial gradient) and the frame difference signal (temporal gradient) [13, 14]. Pel-recursive and differential-based techniques start with the assumption that the sum of the spatial and temporal derivatives of the image intensity remain constant.

Horn and Schunck

From Horn and Schunck [15], assume that the intensity of an imaged point is constant:

$$\frac{d\mathcal{I}}{dt} = 0 \tag{3.1}$$



(a) (b)

Figure 3.5: *Aperture problem: (a) has no intensity variation so only motion normal to the translating straight contour can be determined, (b) has intensity variations so motion can be correctly determined given suitable assumptions.*

where $\mathcal{I}$ is the instantaneous image and t is the time. Application of the chain rule for differentiation gives:

$$\mathcal{I}_m = \frac{\partial \mathcal{I}}{\partial x}u + \frac{\partial \mathcal{I}}{\partial y}v + \frac{\partial \mathcal{I}}{\partial t} \tag{3.2}$$

where $u = \frac{dx}{dt}$ and $v = \frac{dy}{dt}$ are the horizontal and vertical velocity components. The second and higher order terms become zero as $\delta t \rightarrow 0$. This is normally referred to as the Motion Gradient Constraint or the Optical Flow Constraint [4, 16, 17] and is referred to as $\mathcal{I}_m$. It is possible to estimate the spatial derivatives $\frac{\partial \mathcal{I}}{\partial x}$ and $\frac{\partial \mathcal{I}}{\partial y}$ and the temporal derivative $\frac{\partial \mathcal{I}}{\partial t}$ using, for example, the central difference method [18]. However, it can be seen that the problem of solving 3.2 is ill-posed because there are two unknowns (u,v) and only one constraint. The solutions are constrained to a line in velocity space instead of a single motion vector; this is known as the *aperture problem*. As a physical analogy, it is not possible to detect the true motion of a line that is visible through an aperture but whose end points are outside the enclosed field-of-view. Given no intensity variation (a straight line) it is only possible to determine motion perpendicular to the line. However, if there are some intensity variations (a curved line) then it is possible to determine motion given certain assumptions. This is illustrated in Figure 3.5.

Horn and Schunck introduce a Smoothness Constraint, referred to as $\mathcal{I}_s$, which states that the velocity field of the intensity patterns in the image varies smoothly and so Equation 3.3 must be small.

$$\mathcal{I}_s^2 = \left(\frac{\partial u}{\partial x}\right)^2 + \left(\frac{\partial u}{\partial y}\right)^2 + \left(\frac{\partial v}{\partial x}\right)^2 + \left(\frac{\partial v}{\partial y}\right)^2 \tag{3.3}$$

Therefore, the error function E that must be minimized to find the motion field is:

$$E = \mathcal{I}_m^2 + \lambda \mathcal{I}_s^2 \tag{3.4}$$

where λ acts as a weight between $\mathcal{I}_m$ and $\mathcal{I}_s$ with larger values leading to a smoother motion field. The choice of λ varies from 0.5 to 100 in the literature [12, 15], but should be proportional to the level of image noise.

Horn and Schunck propose the following iterative solution to Equation 3.4:

$$u^{n+1} = \bar{u}^n - \frac{\mathcal{I}_x(\mathcal{I}_x\bar{u}^n + \mathcal{I}_y\bar{v}^n + \mathcal{I}_t)}{\lambda + \mathcal{I}_x^2 + \mathcal{I}_y^2} \tag{3.5}$$

$$v^{n+1} = \bar{v}^n - \frac{\mathcal{I}_y(\mathcal{I}_x\bar{u}^n + \mathcal{I}_y\bar{v}^n + \mathcal{I}_t)}{\lambda + \mathcal{I}_x^2 + \mathcal{I}_y^2} \tag{3.6}$$

where u^n and v^n are the motion horizontal and vertical estimates, respectively, at iteration n for a particular pixel and $\bar{u}^n$ and $\bar{v}^n$ are the average motion estimates around the pixel. The partial derivatives of $\mathcal{I}$ for x, y and t are given as $\mathcal{I}_x$, $\mathcal{I}_y$ and $\mathcal{I}_t$, respectively. For a single solution between two frames this scheme must iterate until the solution stabilises, which may be upwards of a 1000 iterations. Alternatively, in an image sequence only one iteration could be performed per frame given an initial good guess from the previous frame.

Alternative Constraints

Singh and Allen [19] generalise image-flow information into two categories: conservation information and neighbourhood information. They divide current optical flow algorithm constraints into these two categories; from a differential optic-flow perspective these are the two constraints $\mathcal{I}_m$ and $\mathcal{I}_s$, respectively. Alternative neighbourhood constraints have been proposed [20], including local constraints that aim to reduce computational cost of the original Smoothness Constraint ($\mathcal{I}_s$). Little and Verri [18] studied a number of different constraints and concluded that local constraints reduce the computational cost without affecting the accuracy of the results, under most conditions.

Another problem with the Smoothness Constraint ($\mathcal{I}_s$) is that it is a quadratic regularisation method and is over regular, tending to blur motion discontinuities. In the presence of multiple motions, non-quadratic regularisation methods have been presented that eliminate outliers and maintain sharp motion discontinuities [21]. These can have a significant improvement on accuracy given multiple motions.

An alternative solution to the aperture problem is to assume that both first-order and second-order derivatives of (3.1) are constant. Expansion of (3.1) yields:

$$\begin{bmatrix} \frac{\partial \mathcal{I}}{\partial x} & \frac{\partial \mathcal{I}}{\partial y} \\ \frac{\partial^2 \mathcal{I}}{\partial x^2} & \frac{\partial^2 \mathcal{I}}{\partial x \partial y} \\ \frac{\partial^2 \mathcal{I}}{\partial y \partial x} & \frac{\partial^2 \mathcal{I}}{\partial y^2} \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = - \begin{bmatrix} \frac{\partial \mathcal{I}}{\partial t} \\ \frac{\partial^2 \mathcal{I}}{\partial x \partial t} \\ \frac{\partial^2 \mathcal{I}}{\partial y \partial t} \end{bmatrix} \tag{3.7}$$

Removing the Motion Gradient Constraint ($\mathcal{I}_m$) Uras *et al.* [22] produce a second constraint equation based on the second-order derivatives:

$$\mathbf{H}\mathbf{d} = -\nabla \mathcal{I}_t \tag{3.8}$$

where $\mathbf{H}$ is the Hessian (with respect to the spatial coordinates) of $\mathcal{I}$, $\mathbf{d}$ is the motion vector $(u, v)^T$ and $\mathcal{I}_t$ denotes the partial temporal derivative. Motion can be recovered as long as $\mathbf{H}$ has an inverse (i.e. Det($\mathbf{H}$) is non-zero).

Second-order techniques can, in theory, produce very precise measurements but they depend on accurate numerical second derivatives, leaving them susceptible to noise in the image. Techniques to identify and extract correct measurements are normally applied, producing a sparse measurement field of accurate measurements. Only if a dense measurement field is required are second-order techniques unsuitable.

Parameter Choice

The choice of differential optical flow parameters is complex and often empirical with authors disagreeing over choice. For example, with Horn and Schunck's technique [15] the choice of the number of iterations and of λ can dramatically effect the performance on a given data-set [12]. Ng and Solo [23] recently proposed new schemes to identify optimal parameters for a local regularisation in optical flow. However, few extensive comparisons have been performed due to the prohibitively large array of techniques.

Liu *et al.* [24] investigate the effect of parameter choice on the accuracy and efficiency in optical flow. They treat this as a direct trade-off and produce *accuracy-efficiency* curves for parameter choice. In general this presents a useful design tool, although some parameters do not fit this model and the *accuracy-efficiency* curve can be altered through the use of specialist hardware, for example Bülthoff [25] developed a massively parallel implementation of optical flow.

3.3.2 Correlation-based Optical Flow

Correlation-based optical flow techniques group the motion of local pixels into regions, whose motion can be considered together [26]. The motion of the region is estimated by limiting the motion to a maximum displacement η in any direction and searching for the most similar region in subsequent frames. The choice of similarity measure, search algorithm and region shape have a strong effect on the accuracy and the efficiency of the technique.

It is very important when considering correlation-based techniques to remember the distinction between motion tracking and motion compensation. Many correlation-based techniques have been developed for video coding applications where the underlying performance criteria is that the reconstructed image is visually acceptable to the user. In this case, while it may be advantageous to get the true motion of a region, it is not necessary. Essentially, the accuracy of

the motion estimation remains secondary to the visual quality of the reconstructed image.

Similarity Measures

The standard similarity measures are a summation of a pixel by pixel comparison between the isomorphic regions. The generic formulation for a square block of N pixels is:

$$D = \sum_{x=0}^{N-1} \sum_{y=0}^{N-1} f\left(\mathcal{I}\left(\mathbf{x}, t\right), \mathcal{I}\left(\mathbf{x} + \mathbf{d}, t + \Delta t\right)\right) \tag{3.9}$$

where $\mathbf{x} = (x, y)^T$ and $\mathcal{I}(\mathbf{x}, t)$ is the intensity of the pixel at position $\mathbf{x}$ at time t. A series of motion vectors $\mathbf{d} = (u, v)^T$ are examined to find the position of best match.

Two commonly used similarity measures are the Sum of Absolute Differences (SAD), $f(r, s) = |r - s|$, and the Sum of Squared Differences (SSD), $f(r, s) = (r - s)^2$. Both measures are computationally simple but the Sum of Absolute Difference does not require any multiplication making it highly amenable to hardware implementation.

Gharavi and Mills [27] proposed the Pel Difference Classification where each pixel in the region is classified as either an exact *match* or *mismatch*. The region within the search area that has the most number of matching pixels is considered the position of best match. However, under large amounts of noise the number of mismatches rapidly increases, in particular the Pel Difference Classifier does not distinguish between blocks that are structurally identical but have undergone a scene illumination change.

Video coding applications have led to the development of reduced bit variations of many of the standard similarity measures. Instead of the full intensity resolution being used, normally 8-bits, only the top n-bits are employed in the comparison. In hardware a 4-bit or 6-bit SAD operation can present a significant saving but this does not apply to general-purpose processors, which now perform 8, 16 and 32-bit operations as standard. The number of bits used in the comparison is reduced with a clear trade-off between accuracy and efficiency [28, 29].

In video coding applications a change in image intensity is important but in motion tracking the target may undergo intensity change, such as lighting variations, which must be ignored for a successful track. Another set of similarity measures are based on the correlation coefficient

and the normalized correlation coefficient (Pearson's R in [30]) has the advantage of being insensitive to image intensity changes. It is given below, with $\mathcal{I}(\mathbf{x}, t)$ as the image intensity at position $\mathbf{x} = (x, y)^T$ at time t, $\overline{\mathcal{I}_t}$ as the average image intensity at time t and $\mathbf{d} = (u, v)^T$ as the motion vector currently being examined:

$$ZNCC = \frac{\sum_x \sum_y \left(\mathcal{I}\left(\mathbf{x}, t\right) - \overline{\mathcal{I}_t}\right)\left(\mathcal{I}\left(\mathbf{x}+\mathbf{d}, t+\Delta t\right) - \overline{\mathcal{I}_{t+\Delta t}}\right)}{\left[\sum_x \sum_y \left(\mathcal{I}\left(\mathbf{x}, t\right) - \overline{\mathcal{I}_t}\right)^2 \times \sum_x \sum_y \left(\mathcal{I}\left(\mathbf{x}+\mathbf{d}, t+\Delta t\right) - \overline{\mathcal{I}_{t+\Delta t}}\right)^2\right]^{\frac{1}{2}}} \tag{3.10}$$

Ordinal, or rank correlation measures, provide another set of similarity measures. Here the calculation is based not on the pixel-intensity but on the rank matrices for the regions being compared. Bhat and Nayar [31] demonstrate that ordinal measures can outperform linear similarity measures in the presence of data outliers. However, under Gaussian noise the ordinal measures perform poorly in comparison to linear similarity measures.

Chen *et al.* [32] propose a generic adaption to similarity measures that weights the smoothness of the current motion vector, in comparison to the motion of the neighbouring pixels, with the output from the similarity measure. This neighbourhood information is actually a form of smoothness constraint [19] but it does not affect the underlying accuracy of the similarity measure. Further similarity measures have been proposed to address application specific issues, such as stereo matching in image registration problems [33].

In motion tracking literature, the SAD, SSD and ZNCC are often used interchangeably with no explicit reason given. In hardware implementations, the SAD is an obvious choice due to the lack of multiplication. However, it should be noted that from these three similarity measures the ZNCC is theoretically optimal under gaussian white noise. Furthermore, depending on the application, particularly the level of image noise, the performance of the three similarity measures may vary significantly. Therefore, it is important to assess the choice of similarity measure even though literature commonly does not provide a detailed analysis.

Search Algorithm

The Full Search Algorithm (FSA) is the simplest search algorithm and performs an exhaustive search through all possible motion vectors u at time $t + \delta t$. Given a maximum velocity the

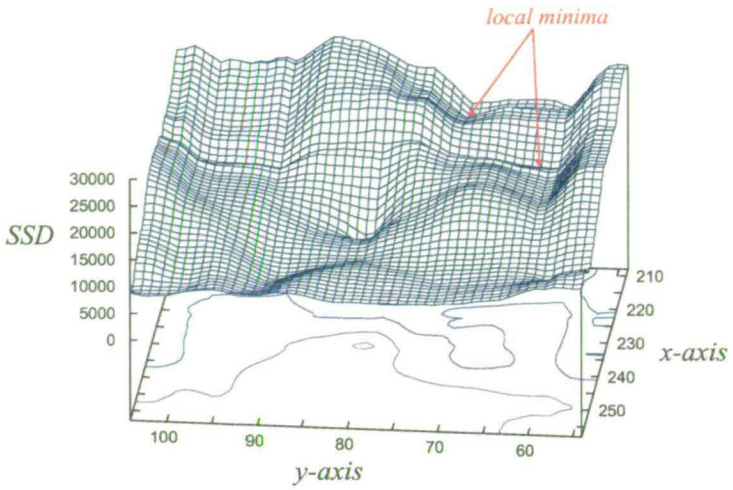


Figure 3.6: *Error surface showing multiple local minima contrary to the UESA; global minima found at (82,235). Computed from frame 16 of the Cyclist data sequence (Section 3.2) using the SSD similarity measure.*

search is restricted to a search radius ω. Normally the search radius would be a square of size $n \times n$. The FSA is computationally expensive and is often too slow for practical implementation. There has been much work done to improve the computational performance over that of the FSA, normally focused on a coarse to fine refinement of the correlation surface. The Unimodal Error Surface Assumption (UESA), where the correlation surface is assumed to rise monotonically around the position of best match, is commonly used [34, 35] to refine an initially coarse search. However, the UESA cannot be guaranteed and, in practice, is often found to be incorrect (Figure 3.6), leading algorithms to be trapped in local minima.

The Three-Step Search (TSS) and its derivatives are amongst the most popular search algorithms, particularly in low bit-rate video coding applications, owing to their effectiveness and simplicity. The original TSS proposed by Koga *et al.* [36] is based on a uniformly allocated search pattern but it is not very efficient when dealing with stationary or quasi-stationary blocks. Li *et al.* [37] observe that in video coding applications the distribution of motion vectors is typically centre-biased so they propose the New TSS (NTSS) which introduces the *halfway-stop* criteria giving faster performance with small motion vectors. The Four-Step Search, proposed by Po and Ma [38], lowers the computational cost at a slight impact on the accuracy of the NTSS. Other derivatives have been presented [39] that show improvements in both performance and cost of computation when compared with the TSS.

The Correlation-Feedback Technique proposed by Pan, Shi and Shu [40] uses the motion vector

from the previous time frame to adapt the initial search pattern. While the Correlation Search proposed by Tsai *et al.* [41] uses both spatial and temporal correlation of motion vectors to determine if any neighbour motion vector is good enough to take as the current motion vector. Otherwise a conventional search algorithm is used to obtain the motion vector.

The successive elimination algorithm proposed by Li and Salari [42] limits the number of search positions for the Mean Absolute Difference to those whose sum norms (R) satisfy the inequality:

$$R - \mathrm{MAD}(m,n) \leq \mathrm{M}(x,y) \leq R + \mathrm{MAD}(m,n) \tag{3.11}$$

where $\mathrm{M}(x,y)$ is the sum norm of a block with motion vector (x,y). The $\mathrm{MAD}(m,n)$ is the Mean Absolute Difference at the position of best match (m,n) identified up to this stage of the search. This produces a reduction in computational cost of around 85% in comparison to the FSA. Lee and Chen [43] propose a technique based on the block sum pyramid which also reduces the computational cost by adapting the calculation of the MAD.

Camus [44] demonstrates that the search can be performed linearly in time (over a number of frames t) while the search area in space N is held constant. In practice this mathematical rearrangement does not provide a performance improvement on standard techniques but it does allows sub-pixel accuracy with a degree of quantization.

Many of the fast search algorithms proposed are sub-optimal and may get trapped in local minima, leading to an incorrect motion estimate. The algorithms focus on achieving small improvements in either the PSNR or computational performance, with many papers using these as the basis for a comparison between techniques. Decroos *et al.* [45] propose a new search algorithm, from a video surveillance perspective, that introduces search points between local minima instead of assuming a single minimum in the search range helping to avoid the possibility of being trapped in a local minimum.

Region Choice

The simplest region choice is a square block of pixels, in video compression usually a block of 8×8 or 16×16. The size of the square block is a trade-off between accuracy and efficiency.

A larger block gives more information for matching but increases the chance that the block will span a motion discontinuity. Oh and Lee [46] propose an adaptive system which groups neighbouring blocks with similar motion vectors into one large block for the next frame. Chan *et al.* [47] propose splitting the blocks based on the minimum sum-of-squared errors, but also include a merge step to allow for complex structures. Malo *et al.* [48] investigate splitting criteria based on spatial and spectral entropy measures. Recently techniques have been introduced or extended to cope with the arbitrary-shaped video object plane coding standards [49, 50] that use an alpha mask to define an arbitrary region shape.

Spatial Transforms

While techniques such as arbitrary-shaped video object plane coding standards allow any shape of object to be tracked, the motion model is purely translational. More complex motion models have been proposed which apply spatial transforms to the region. Sullivan and Baker [51] propose a motion compensation method based on Control Grid Interpolation. Their results show a significant reduction in block artefacts compared to standard block matching, however, the computational cost is very high. Sefredis and Ghanbari [52] investigate affine, perspective and bilinear motion models and, in the context of video coding, produce significant improvements but at a large increase in computational cost. Nakaya and Harashima [53] demonstrate that the optimal region shape for bilinear transformations is a square block and a triangle for affine transformations. Lopes and Ghanbari [54, 55] have recently demonstrated improvements using spatial transforms with overlapping and hierarchical techniques for video coding applications.

3.3.3 Frequency-based Optical Flow

Frequency-based optical flow techniques operate in the Fourier domain and use orientation sensitive filters to detect motion. However, they require a significant overhead in computation with common techniques [56, 57] requiring banks of Gabor filters. More recently, it has been noted [58] that the phase rather than the amplitude should be employed because phase is relatively insensitive to contrast variations. However, a bank of fourier-domain filters is still required making it computationally very expensive.

Another set of frequency-based methods uses Discrete Cosine Transform (DCT), or psuedophase, instead of the Fourier transform. Koc [59] proposed a psuedophase method that ex-

hibited good performance under noise. However, it utilises the 4 coefficients from a 2D DCT coder making it suitable for video coding applications where these are already available but computationally intensive otherwise.

3.3.4 Optical Flow Extensions

Hierarchical Approaches

Without *a priori* knowledge of the image motion, a large search radius may be required to detect all possible motion in the image. Hierarchical approaches detect motion at a reduced resolution and then refine the estimate at increasing resolution. Glazer [60] proposed a hierarchical approach based on Horn and Schunck [15]. Anandan [61] proposed a hierarchical scheme based on an overlapping Laplacian pyramid image structure that refines the motion estimates from low to high resolution of the pyramid, at each stage passing a confidence measure for the motion estimate down to the next stage. The approach is compatible with correlation and differential optical flow techniques, although Anandan used correlation-based approaches.

Colour Invariance

Golland and Bruckstein [62] extend the concept of intensity invariance in an image to colour invariance for differential optical flow, using the RGB and HSV colour spaces. They show improvements over intensity invariance as long as their assumptions on the scene illumination and object reflectivity properties hold. Magarey *et al.* [63] introduces chrominance information using a noise-decorrelating colour space transform, which optimizes the amount of colour information available. Konrad and Dubois [64] use the *Maximum a Posteriori Probability* to produce a three-term minimization problem based on any vector colour model.

Within correlation-based techniques, Wei and Li [65] used an estimate of the image illumination to compensate the MAD calculation between frames. However, this only produces an increase in accuracy in a small number of situations. Koschan *et al.* [66] linearly combine the three colour channels but, again, produce limited improvement for a three-fold increase in computational cost. Mecke *et al.* [67] produce a motion estimate from a weighted sum of the three individual (RGB) estimates. The weights are a confidence measure based on the surface of the MAD for each colour channel. This demonstrated a better improvement in accuracy, which partially offsets the three-fold increase in the computational cost.

While techniques based on colour information can produce improvements under certain circumstances, they often utilise strong assumptions about the lighting model and scene structure that do not hold under all circumstances and contribute little when there is limited colour information in the scene. Furthermore, they provide a significant increase in computational cost from the original algorithm.

3.3.5 Feature Correspondence

Feature correspondence techniques identify sets of features in frames and attempt to match the corresponding features between adjacent frames giving a series of point motions, or a correspondence map. The motion between frames can then be calculated by fitting a motion model to the correspondence map.

Feature correspondence has been widely applied to Motion Estimation, in fields as diverse as outdoor vehicle navigation [68] and face tracking for video-conferencing techniques [69]. Feature correspondence can also be used in image registration techniques where there are two images taken at, for example, different times, from different sensors or from different viewpoints [70]. Feature correspondence allows significant complex motion to be present between the frames. However, the difficulty of the registration often leads to hybrid techniques such as Zheng and Chellappa [71] who obtained robust registration of satellite images but at a significant computational cost.

Feature Selection

While all stages are important, the accurate identification of features is vital. Therefore, selecting features that can be tracked accurately through the sequence is an important but difficult task and no single feature type will be suitable for all applications. Researchers have proposed tracking many different features including regions, surfaces, edges and corners [11].

The kind of features used depends mostly on the application. For instance, edge segmentation has been widely used in indoor navigation [72], where doors and corridors are common. Corners features computed from the raw greyscale images have also been widely used [73] and are better suited to more general scenes. While region features are often used in areas of low structure but high texture [74].

Unusual features have been proposed for specific tasks. These include complex features such as tracking known facial features under six-degrees of freedom [75] or tracking planting patterns of flowers for an agricultural robot [76]. Seemingly abstract features have also been proposed, such as tracking the point of intersection between lines of symmetry between edges in the image, which is based on psychophysical experiments for robot gaze-fixation [77]. However, these features are highly application specific and not useful to a wider audience.

In general, areas of low texture or areas of repetitive texture can present problems for feature identification. In optical flow the high spatial density of the motion estimates can be used to compensate for poor areas. Horn and Schunck [15] used the smoothness constraint to this effect or, more generally, Singh and Allen [19] referred to this as neighbourhood information. In feature correspondence it is not possible to utilise neighbourhood information.

To overcome this, researchers have proposed to track features with sufficiently high spatial content [78] or regions where a combination of second-order derivatives meets a given threshold [71]. However, while these minimize the number of bad features, even a region rich in texture can be a poor choice. For example, it may straddle a depth discontinuity or be dependent on current scene illumination. In both cases the feature is not a fixed point in the world but a product of the current scene. An external measure, either to indicate confidence in the positional accuracy of the feature match or to threshold features that have undergone large amounts of change over the image sequence, can be valuable in improving accuracy of the motion estimate. Shi and Tomasi [79] proposed a dissimilarity measure for this purpose.

Another problem for feature selection is that over time the appearance of the features being tracked will alter, particularly if they undergo non-translational motion. Many of the proposed features are difficult to extend to non-translational motion [79]. Methods that are either Affine or Similarity transform invariant have been proposed; including globally defined invariants, such as moment invariants [80] and Fourier descriptors [81], and locally defined invariants, such as curvature derivatives [82]. Techniques that use local invariants can be more precise but are sensitive to noise in the image. While techniques that use global invariants are susceptible to feature occlusion but tend to be more robust and less sensitive to noise. More recently Chen *et al.* [83] proposed a novel affine invariant feature set that did not require the high-order derivatives of curvature derivatives but whose semi-local computation claims to avoid many of the occlusion problems associated with global invariants. It is defined as semi-local because the mask size can be altered. In practice this offers a trade-off within the algorithm between global

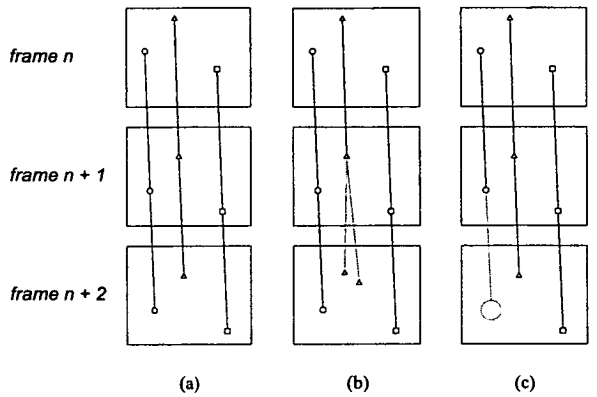


Figure 3.7: *Three moving points are measured at three time instances. (a) shows the classic motion correspondence problem while (b) and (c) demonstrate two degenerate situations that can cause considerable difficulty. In (b) a new feature has been detected in the final frame close to a previously tracked feature. In (c) a tracked feature has not been detected in the final frame.*

and local invariants but does not fundamentally address the problems associated with either.

Motion Correspondence

Given two sets of simple features from consecutive frames it is necessary to determine the correspondence. The complexity of this task varies but if the motion between frames is considerably smaller than the distance between features then the problem is much simpler. It is possible to enforce a minimum distance between features through a check in the feature selection stage. However, in situations where the position of features cannot be easily dictated or where the motion between frames is very large a more complex correspondence technique may be required.

Veenman *et al.* [84] present an overview of previous techniques and propose a new algorithm, which demonstrates good results under dense, fast-moving sets of features. Instead of using only the position information they employed velocity state information as well to predict likely targets. They also split techniques into into statistical method [85] and heuristic methods [86]. However, these are simply different ways of using the position and velocity information to obtain a best-fit solution to the problem.

Figure 3.7 shows the classic motion correspondence problem with two degenerate situations. It is likely that some features will move outside the field-of-view, be subject to occlusion and disocclusion or simply not be detectable. This is unavoidable and the motion correspondence

stage must be robust to these issues.

Region Correspondence

Region correspondence can be considered a special case of correlation-based optical flow where a sparse motion field is calculated. All the techniques presented in the previous section on correlation-based optical flow are relevant (Section 3.3.2). Unlike simple features, such as edges, there is no correspondence problem. Instead, the key stage is identification of the regions to be tracked and a number of techniques have been presented. Many region choice techniques are based on the concept of the aperture problem, attempting to identify regions with high spatial gradients in both directions [78].

A common feature correspondence technique for motion estimation is the Kanade-Lucas-Tomasi (KLT) algorithm [87] that tracks region features. Starting with Equation 3.1 they state:

$$\mathcal{I}(\mathbf{x} - \mathbf{d}, t) = \mathcal{I}(\mathbf{x}, t + \Delta t) \tag{3.12}$$

and apply a SSD function to produce the residual:

$$\epsilon = \sum_{\mathcal{W}} [\mathcal{I}(\mathbf{x} - \mathbf{d}, t) - \mathcal{I}(\mathbf{x}, t + \Delta t)]^2 \tag{3.13}$$

were $\mathcal{I}(\mathbf{x}, t)$ is the image intensity at position $\mathbf{x} = (x, y)^T$ at time t, $\mathbf{d} = (u, v)^T$ is the motion vector being tested and $\mathcal{W}$ is the region over which this is computed. When the motion vector $\mathbf{d}$ is small Equation 3.12 can be approximated by a Taylor series truncated to just the linear term:

$$\mathcal{I}(\mathbf{x} - \mathbf{d}, t) = \mathcal{I}(\mathbf{x}, t) - \mathbf{g} \cdot \mathbf{d} \tag{3.14}$$

Therefore, we can write the residual as

$$\epsilon = \sum_{W} [\mathcal{I}(\mathbf{x}, t) - \mathcal{I}(\mathbf{x}, t + \Delta t) - \mathbf{g} \cdot \mathbf{d}]^2 \tag{3.15}$$

By substituting the first-order Taylor series into the SSD residual and imposing a constraint that the derivatives with respect to $\mathbf{d}$ are zero, they obtain a final system to be solved:

$$G\mathbf{d} = e \tag{3.16}$$

where $G = \begin{bmatrix} g_x^2 & g_x g_y \\ g_x g_y & g_y^2 \end{bmatrix}$ and $e = \sum_W [\mathcal{I}(\mathbf{x} - \mathbf{d}, t) - \mathcal{I}(\mathbf{x}, t + \Delta t)] \begin{bmatrix} g_x \\ g_y \end{bmatrix}$.
This system is normally solved using a Newton-Raphson iteration method, which in practice achieves good accuracy within a small number of iterations.

Shi and Tomasi [79] investigated a feature selection algorithm using constraints placed on the minimum value for the Eigenvalues for the matrix G and reported good tracking using the KLT tracker. Calvagno *et al.* [69] extended Kanade-Lucas-Tomasi to include feature updating through the sequence and applied this to face tracking. Tommasini *et al.* [88] demonstrated a robust version of Kanade-Lucas-Tomasi that used an outlier rejection scheme without excessive additional computational cost.

3.4 Passive Navigation and Motion Filtering

The introduction divided the problem of ego-motion estimation into two separate stages:motion estimation and passive navigation. The first stage attempts to calculate motion for different parts of the image, either on a per-pixel, per-region or per-feature basis, between two frames. This has been addressed in the previous section. It is then necessary to segment and integrate these into a single motion estimate for the underlying objects, which has been referred to as passive navigation. One way of looking at this is a mathematical regression problem where numerous data points (individual motion estimates) must be made to fit into a model (perspective transform). However, it is important to remember that the purely image-based (pixels) motion estimation must now be translated into real-world motion (millimetres). It is at this stage that

many external factors, such as the scene depth and camera set-up, become important.

There are currently many different research areas including 3D structure-from-motion [89], 3D terrain topography for vehicle guidance [90, 91] and underwater navigation. In the majority of real situations the scene will have considerable depth variations and the use of stereo images has been popular. However, for downward looking vehicle navigation, the depth variations are relatively small and the extra information provided from a second camera would be limited. For this reason, the stereo or depth based techniques are not discussed in detail. Similarly structure-from-motion and terrain topography techniques, which are major areas of research in themselves, are not reviewed. Instead, a general review of motion tracking is given, with focus on passive navigation for vehicle navigation. In some cases, where the motion is primarily planar 2D it is possible to perform a least-squares fit to a homography (mapping between points). This homography can then be translated into real-world motion. The same fundamental techniques are applied towards the same basic goal of finding the motion described by the individual motion estimates.

3.4.1 Passive Navigation for Vehicle Navigation

Passive navigation has been an active area of research since the late 1970's. A taxonomy for passive navigation is difficult due to the many techniques currently employed. An early taxonomy was to split the work into three categories: discrete, differential and continuous [92]. Unfortunately, it is currently difficult to classify many of the modern techniques with these categories. Instead, the different classes identified in Tian *et al.* [93] are used in this section. The five different underlying principles they identified are: Elimination [92], Motion Parallax [94], Image Deformation [95], Rotation First [96] and Epipolar [97].

Although other papers had addressed the problem of passive navigation, Bruss and Horn [92] were among the first to present a numerically robust continuous solution to the problem. This work was later covered in greater detail in [98]. They proposed to recover the motion of a camera with respect to a planar surface using a least-squares technique. They remove depth algebraically from the equation but the instantaneous motion recovery problem remains ill-posed so they must also assume the planar surface motion is rigid. This is a particularly strong assumption to make; assuming it is not broken, it provides good structure to the estimate [93]. They encountered a difficulty in obtaining a solution that was linear in all the motion parameters. Their best solution was to approach the minimization using the $ML_{\alpha\beta}$ norm (Equation

3.17). This weights the error contributions, giving greater importance to points with larger velocity. This is appropriate when the measurement of larger velocities is more accurate. The use of the $ML_{\alpha\beta}$ results in three linear parameters (rotation) and three non-linear parameters (translation). For this reason, this method requires a non-trivial numerical solution. The $ML_{\alpha\beta}$ norm is defined as:

$$\| f(x,y) \|_{\alpha\beta} = \iint f(x,y)^2(\alpha^2 + \beta^2)dxdy \tag{3.17}$$

given

$$\alpha = -U + xW$$
$$\beta = -V + yW$$

where $(U, V, W)^T$ is the translation vector and (x, y) are the projected coordinates.

Negahdaripour and Horn [99] adapted this work to allow direct computation of the motion from image brightness gradients, avoiding the computation of optical flow as an intermediate step. They also demonstrated that a closed-form solution was possible, avoiding the complex numerical techniques. Complete derivations can be found in the two technical reports [100, 101]. They demonstrate accurate results in their testing, although the testing is rather limited. Negahdaripour and Yu [102] went on to address some of the inherent ambiguities in passive navigation but one problem that remains is the existence of a dual-solution, due to the presence of a squared term. Although in practice, the second solution can be calculated from the first.

Motion parallax techniques, such as Heeger and Jeepson [94], note that the difference between any two optical flow vectors gives a constraint on translation that is independent of rotation. A set of orthogonal constraint vectors is then constructed and the estimate of T can then be found. The advantage of this technique is that T is computed directly, avoiding any numerical technique but it does not utilise as much information as Bruss and Horn's [92] rigidity constraint.

Tomasi and Shi [95] utilised image deformations, which can be derived from motion paral-

lax information. They defined the deformation as the angular change between pairs of image points as the camera moves and they use this to estimate $\mathbf{T}$ independent to rotation. However, the major disadvantage is that for a large number of points the implementation is much more computationally expensive than many alternatives. Tian *et al.* [93] also report that the Tomasi and Shi algorithm [95] often fails to converge.

The previous techniques attempt to identify translation first using rotation invariant constraints. Instead, Prazdny [96] derives an algebraic expression that is independent of translation and depth. This expression can then be solved using a triple of image points. However, it requires the minimization of three third-order polynomials and has proven, in extensive tests, to be highly sensitive to noise [93].

The final set of techniques is based on the epipolar constraint, or coplanarity constraint. Two camera coordinate systems can be related by a rotation $\mathbf{R}$ and a translation $\mathbf{T}$:

$$\mathbf{x}' = \mathbf{R}\mathbf{x} + \mathbf{T} \tag{3.18}$$

The epipolar constraint states that all three components of this equation ($\mathbf{x}'$, $\mathbf{R}\mathbf{x}$, $\mathbf{T}$) are coplanar. Kanatani [97] reported an instantaneous time method to recover image motion using this constraint. Kanatani also reported a statistical bias and proposed a suitable renormalization technique to improve the accuracy. However, there are non-rigid motions that satisfy the epipolar constraint making this a weaker constraint than the rigidity constraint [92]. Tian *et al.* [93] reported that the Kanatani's technique did not perform as well in extensive simulations as alternative techniques and speculated that the weak constraint is responsible.

Further alternatives, include Branca *et al.* [103, 104] who attempt to calculate only the heading-direction and time-to-collision, which they argue are the two most important navigation parameters. It is necessary to add external constraints but Silva and Santos-Victor [105] investigated how best to minimize the effect of these constraints without affecting the performance of the algorithm.

The survey by Tian *et al.* [93] used a representative technique from each of the classes defined. Bruss and Horn [92] consistently produced the most accurate results. However, the required numerical minimization exhibited convergence problems, particularly under sideways motion.

A good initial guess can help to alleviate this problem. One final point is that it is often reported in the literature that an increase in the field-of-view will increase accuracy. However, Tian *et al.* [93] report that any increase in field-of-view must be matched by an increase in image resolution. Given that an increase in image resolution is often impractical a difficult trade-off exists between these two factors.

Robust Regression and Outlier Rejection

Even using a dense set of optical flow measurements, least-squares (LS) strategies and similar unweighted techniques are sensitive to errors in the optical flow. In classical theory, the errors are assumed to be normally distributed with a zero mean and unknown standard deviation σ. Classical techniques perform optimally when this model holds[106]. However, real data does not often completely satisfy the classical assumptions, in particular data containing outliers. The two approaches to this problem are outlier diagnostics and robust regression. Outlier diagnostics attempt to identify and remove data outliers before performing the regression while robust regression performs the regression while minimizing the effect of outliers on the result. Investigation has proven that classic least-squares techniques can be inadequate when applied to passive navigation [107] and robust regression estimators have been investigated: the two most common classes are S-estimators and M-estimators.

An M-estimator replaces the squared residual function, r_i^2, in LS with another function of residual of the form

$$\text{Minimize} \sum_{i=1}^{n} \rho\left(r_i\right) \tag{3.19}$$

where ρ is a symmetric function with a unique minimum at zero [106]. However, the solution can be expensive and it is still vulnerable to leverage points (errors that shift the principal axis of the least-squares regression, Figure 3.8). Generalised M-estimators have been proposed to address this. They introduce a weight function to Equation 3.19 to bound the influence of leverage points. However, generalised M-estimators and their derivatives have a theoretical maximum on their breakdown point with a maximum of 30% currently achieved.

S-estimators are a generalised term for techniques that were first introduced with the Least Me-

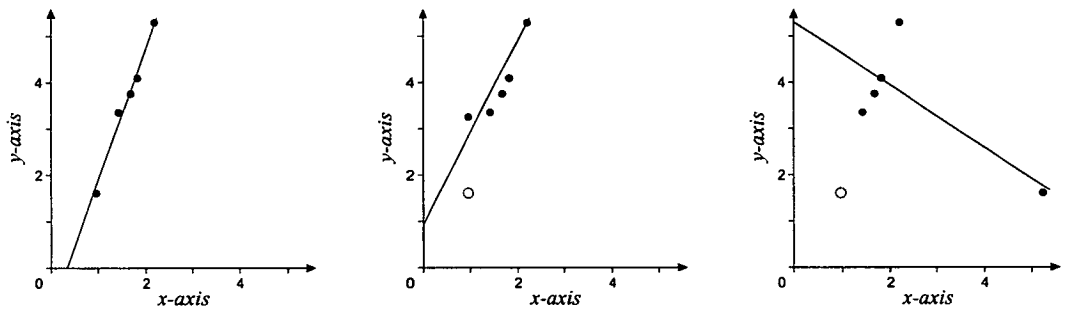


Figure 3.8: *Least square regression line for five data points. (a) shows the data without errors and a good least-squares regression line, (b) has an an error in y-direction but the least-squares fit remains acceptable, (c) has an error in the x-direction (leverage point) which causes the principal axis of the least-squares fit to alter. Data taken from Rousseuw and Leroy.*

dian Squares, where a robust measure of the scatter of the residuals is minimized [106]. The key advantage of S-estimators is their very high breakdown point of almost 50%, which although rarely used in practice provides an incredible degree of robustness. Least Median Squares have been successfully applied to motion estimation [108], however, the computational cost, even with a suitable optimization technique, increases rapidly with the amount of data [109]. The expensive computational calculation (median) combined with a very slow convergence rate led to the introduction of Least Trimmed Squares, which has a significantly better convergence rate, although overall computation remains expensive. A further extension to Least Median Squares or Least Trimmed Squares is Reweighted Least Squares where a weighted least squares regression is performed based on the identification of outliers, using the robust measure (r_i) from either Least Median Squares or Least Trimmed Squares as the weight.

Many other robust estimators have been proposed in the literature; Rousseeuw [106] and Stewart [110] provide summaries of commonly used robust estimators. Although significant work has been carried out [111, 112] to reduce the computational complexity of the robust regression techniques they still present a significant increase over ordinary LS and Rousseeuw [106] states that ordinary LS techniques are still commonly applied for this reason.

Outlier rejection, previously referred to as outlier diagnostics, attempts to identify and remove outliers before performing the regression. A simple technique is to perform the LS regression and analyse the residuals, removing any points that are outwith a given threshold and redoing the LS regression. This process is repeated until all LS residuals are lower than the threshold. Different rejection rules and their corresponding threshold values, have been proposed in liter-

ature.

A prominent application was by Tommasini *et al.* [113] with the model-free X84 rejection rule. This rule employs median and median deviation instead of mean and standard deviation and proposes to reject any values which are more than k median absolute deviations away from the median. They demonstrate that a value of $k = 5.2$ is appropriate for motion tracking.

Alternatively, Sayrol [114] proposes to map an alpha stable distribution to the correlation error surface. This shows, experimentally, that the alpha stable distribution shows heavier tails under multiple motions, which are a common cause of outliers. However, there is no discussion of any other causes of outliers and their effect on the distribution.

Outlier rejection rules can often be implemented with less computational cost than equivalent robust regression techniques, making them amenable to real-time systems. However, fundamentally LS attempts to minimize the residuals and by definition will minimize the effect of outliers. This may result in a poor fit for the majority inliers by the minority outliers, leading to points being incorrectly identified (Figure 3.8).

An alternative to the robust regression and outlier rejection techniques are classical robust statistics. In particular, maximum likelihood (ML) and maximum a posteriori (MAP) have been successfully applied to motion estimation problems. LS estimators make strong assumptions about the distribution for the random variables (ϵ_i) - normally that they are Gaussian independently distributed with zero mean and common variance. ML estimators turn out to be ordinary LS when these assumptions hold. However, ML estimators provide a generalization when the random variables have a multivariate Gaussian distribution that is more accurate than ordinary LS regression [115]. MAP estimators require that the distribution of the errors and the error parameters are known and that an adequate prior distribution of the parameters is available. It is then possible to use Bayes' Theorem to obtain the posterior distribution and MAP estimates [115] - essentially that given a stronger set of assumptions a more accurate result can be found, if the assumptions hold.

While these techniques can produce improved results under Gaussian noise they are still susceptible to outliers. Sim and Park [116], amongst others, proposed using a robust regression technique, in their case Reweighted Least Squares , within a MAP motion estimation technique. Their results were very encouraging but at considerable expense in computation.

3.4.2 Motion Tracking Filters

The motion estimates from any technique, no matter how robust, will still remain noisy and inaccurate to some degree. This suggests the use of a filter to help minimize the noise present in the output signal. In motion tracking the Kalman filter [117] has been commonly used because a significant improvement in stability and accuracy can be achieved over the raw output. Papanikolopoulos *et al.* [118] demonstrate use of a Kalman filter for visual tracking of a target. Calvagno *et al.* [69] use an extended Kalman filter on top of the Kanade-Lucas-Tomasi feature correspondence technique to improve the stability and accuracy of the final estimate. Mecke *et al.* [67] applied a simple kinematic model of camera motion to the extended Kalman filter to further increase the accuracy of the motion estimates. Another use of Kalman filters has been to perform fusion from different sensors in a natural way [119], although there are many alternatives to sensor fusion. Recently, Nickels and Hutchinson [120] demonstrated that performance degrades if the Kalman filter's implicit assumptions, that the measurements are random vectors with Gaussian PDFs, are violated.

3.5 Motion Estimation for Dervish

All the techniques presented here are mature areas of research and the wider subject of motion in image sequences has been an active area of research since the 1970's. It would, therefore, be incorrect to dismiss any technique as entirely unsuitable. However, the different techniques have their advantages and disadvantages and a careful trade-off must be made. In the case of the Dervish the decision has two unknowns: the hardware platform and the motion estimation algorithm. These two choices are heavily influenced by each other and cannot be separated. The motion estimation algorithm is discussed here, while Chapter 4 examines the Dervish landmine-clearance vehicle and the constraints it places on the navigation system, before discussing suitable hardware platforms.

Many of the real-world attempts at vehicle navigation have focused on fairly general situations, such as off-road all-terrain vehicles or road following with a forward looking camera. In these situations the main challenge is segmentation and understanding of a complex scene, to allow the vehicle to plot a route avoiding the many hazards; DeSouza and Kak [121] present a recent survey on vision for mobile robot navigation. However, the Dervish will already have a route planned out by hand which will avoid any hazards. Instead the challenge for the Dervish is

accurately reporting the vehicle motion over a very short-range.

As previously discussed, an important difference between motion estimation techniques is the density of individual motion estimates. Optical-flow provides a far higher density of information, generally covering the entire image, by utilising certain constraints. Region-based feature correspondence attempts to identify areas of sufficiently high texture to be able to track accurately and independently through the sequence. In our circumstance the requirement is for a single estimation of vehicle motion, suggesting that tracking fewer, higher accuracy regions would be better. Giachetti *et al.* [74] and Mandelbaum *et al.* [122] recently concluded that region-based feature correspondence was suitable for practical vehicle navigation and that its robustness to noise made it the preferred option. Therefore, it was decided that region-based feature correspondence provided the best option.

One major decision with region-based feature correspondence is the search algorithm applied. The Full Search Algorithm is computationally very expensive and two major strategies have been proposed to tackle this: hierarchical search and coarse-to-fine search patterns. The hierarchical search assumes that enough of the structure of a region will be maintained at a reduced resolution to allow a provisional motion estimate that can be refined at a higher resolution. Should an incorrect match be made at a higher level the global minimum may not be found. Similarly coarse-to-fine search patterns employ the UESA and if this is violated (i.e. the error surface is not monotonic and has local minima), the global minima may not be found. It is straightforward to demonstrate a degenerate case for either strategy, for example Figure 3.6 shows an error surface that does not meet the UESA. However, the problems occur for hierarchical motion when the texture required by the correlation is lost at a lower resolution, especially within areas of repeated texture. Unlike the UESA, this is something that can be addressed within the design of the system. Furthermore, the majority of UESA search patterns have been designed for motion compensation and not motion tracking and many of the underlying assumptions, such as adaptive diamond motion [123], would be incorrect. Therefore, the hierarchical search which is common to many region-based tracking schemes, such as Anandan [61] or Kanade-Lucas-Tomasi [87], has been chosen for use. The basic structure of a hierarchical correlation-based feature tracker, using the SSD measure, is shown in Figure 3.9. In this case a two-level pyramid is demonstrated. The use of the hierarchical search is considered further on Chapter 6.

Another important decision is the reference block update strategy. Often a dissimilarity measure

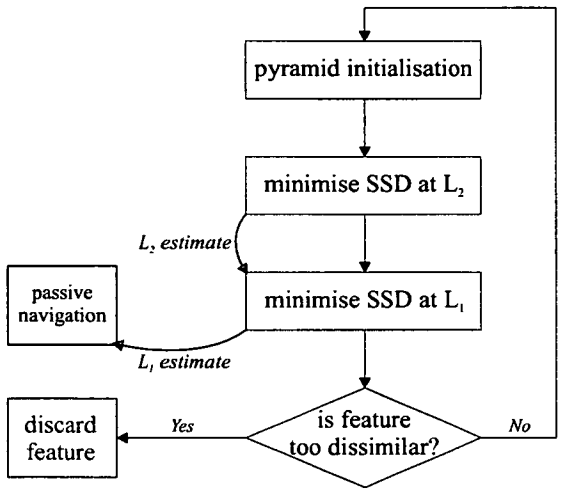


Figure 3.9: *A simplified flow-chart for a correlation-based tracker with a two-level pyramid. The blue curved lines indicate how the system makes use of the motion estimates from each level of the pyramid.*

is used to determine when a region has become too dissimilar to its starting appearance and must be discarded. However, Peacock *et al.* [124] recently proposed some novel reference block update strategies for single-block object tracking, demonstrating a significant improvement in accuracy. An extension to this work is presented in Chapter 4 where a new reference block update strategy is proposed. The application of reference block updating to vehicle navigation is investigated further in Chapter 7.

The individual features tracked must then be combined by the passive navigation system to produce a robust estimate of vehicle motion. Based on the discussion presented in the previous section, the original technique presented by Bruss and Horn [92] seems the most appropriate due the high reliability. However, the projection model can be simplified reducing the degrees of freedom and the overall complexity of the least-squares problem. This structure forms the basis for the motion estimation used in this thesis.

Having reviewed the Dervish Landmine-clearance vehicle and the relevant motion estimation algorithms it can be noted that the large amount of rotation presents a serious problem. Obtaining high accuracy results over an extended period of time when the Dervish rotates approximately twice a minute is a difficult task and this is addressed in detail in Chapter 6. However, the relatively flat terrain and 2D motion means that simpler planar techniques can be employed in place of complex 3D techniques for the passive navigation stage. The exact nature of the motion model is also discussed further in Chapter 6.

3.6 Summary

This chapter has presented a review of techniques for estimating motion within image sequences for vehicle navigation. The process was split into two stages: motion estimation and passive navigation, although some of the techniques can combine both stages. The motion estimation techniques fall broadly into optical flow and feature correspondence; the correlation-based techniques which fit into both categories were discussed within optical flow. Section 3.5 discussed the different motion estimation techniques and selected the most suitable for application to the Dervish landmine-clearance vehicle. The passive navigation techniques have been presented to allow an overall understanding of the estimation of vehicle motion and a suitable technique has been chosen for application to the Dervish landmine-clearance vehicle. However, this technique is not determined to be optimal and alternative techniques may be applicable.

One area that is of recent interest in both motion estimation and passive navigation is robust techniques. Robust techniques have been applied to many areas with some excellent results. In theory, it is desirable to have a robust technique for use in vehicle navigation. However, in practice a trade-off must be made between the computational cost and the performance increase. Given a target of real-time operation on a low-cost platform it is inadvisable to select complex techniques without first confirming that a simpler system will not perform adequately. Furthermore, given the fallibility of all motion estimation algorithms it is necessary to allow redundancy within the system, for example tracking multiple targets, which must be met within a fixed amount of computational power. One area of interest is motion estimation fusion [125] where multiple techniques are fused, allowing not only more robust and accurate estimates but more timely estimates. In practice, the timely estimates would allow a high frame-rate to be maintained, avoiding one of the disadvantages normally associated with robust techniques. Chapter 5 presents a review of suitable low-cost platforms and it can be noted that many multi-chip platforms are available, presenting a good structure for the implementation of multiple motion estimation techniques. Furthermore, the use of correlation-based region tracking provides excellent scalability based on the number of regions tracked providing a flexible decision on the trade-off between accuracy and efficiency.

The motion estimation technique chosen was a correlation-based feature correspondence scheme, utilising a hierarchical search pattern. An important problem within correlation-based techniques is how the reference block is updated and Chapter 4 looks at current methods and goes on to propose a new update strategy. Chapter 5 looks at the hardware platforms for Dervish

and selects a suitable development platform. The motion pattern for Dervish is then analysed in Chapter 6 before the frame-to-frame motion estimation techniques are considered including the application of the hierarchical search pattern. The motion estimation system is then tested for accuracy in Chapter 7 using a feature management system and passive navigation harness. For comparison, Chapter 7 also presents accuracy tests on two well known motion estimation techniques: KLT and Differential Optical Flow.

Chapter 4
Reference Block Updating

4.1 Introduction

An important problem when using a correlation-based feature tracking algorithm is how to cope with the changing target appearance over time. It is common to utilise a static reference block that is never updated. However, a tracked feature may eventually become too dissimilar to the original target to allow reliable tracking or the track may simply have been lost, for example an occluded feature. Shi and Tomasi [79] propose a dissimilarity metric, for which Tommasini [113] developed a robust version. Essentially, if the correlation similarity measure at the position of best match exceeds a threshold then the feature is discarded. A recent alternative, proposed by Peacock *et al.* [124], is to use reference block updating strategies based on either an FIR or Kalman filter. They demonstrated that update filtering could be used to increase the accuracy in a target tracking application. While there is an associated increase in computational cost per feature the increase in accuracy can be significant. In particular, many features that would otherwise have been lost are tracked accurately, thereby extending the useful lifespan of a feature. The reference block update strategies are applicable to any region-based tracking schemes where the target is tracked over an extended period of time or where the target appearance is known to change over time.

The first section of this chapter introduces the test data set used in the investigation of reference block updating strategies. The second section (4.3) of this chapter investigates the performance, both computation and accuracy, of the reference block updating strategies. The parameter choice for each strategy is investigated and performance optimization performed. Section 4.4 then proposes a new reference block update filter based on a least-squares polynomial filter that provides the accuracy and performance of the Kalman filter combined with systematic parameter choice and track-initiation. Finally, Section 4.5 looks at some other problems with reference block updating when applied to real problems. Chapter 6 applies reference block updating to passive navigation and looks at the trade-off, presenting a series of comparative tests to produce a completed algorithm for use with the Dervish landmine-clearance vehicle.

Figure 4.1: *The four general purpose data sets. Clockwise, from the top-left: CarParkB, Cyclist, Pedfollow and Snowseq. The images are not to scale.*

4.2 Test Data Sets

The reference block updating strategies could be applied to any correlation-based motion estimation or motion tracking scheme, therefore, four general purpose data sets have been chosen. For each data set a target was chosen and manually tracked over the sequence, with an accuracy of approximately 1 pixel. These measurements are used as the ground truth when comparing different techniques. All the data sets are available on the attached DVD-ROM along with their manually determined ground truth to allow easy replication of results.

The CarParkB sequence is from a CCTV security camera and follows a car travelling behind a fence. Format conversion has also introduced some artefacts and this sequence is relatively difficult to track. The Cyclist and Pedfollow sequences were taken using a digital video camera and are both relatively noise-free sequences, although the target appearance changes considerably over the sequences. The Snowseq was taken using a digital video camera from an elevated position in poor lighting. The falling snow and tree branches in the scene make target tracking

difficult. CarParkB and Cyclist are grey-scale with a resolution of 720×576. Snowseq is grey-scale with a resolution of 768×576. Pedfollow is grey-scale with a resolution of 384×288. Table 4.1 summarises these sequences and Figure 4.1 shows the first frame from each; the images are not to scale. In the CarParkB sequence the white car is tracked as it moves left along the road with a 16-pixel block. In the Cyclist sequence the cyclists helmet and head are tracked with a 16-pixel block. In the PedFollow sequence the pedestrians head is tracked with a 16-pixel block. Finally, in the SnowSeq the head and shoulders of the pedestrian starting centre-bottom who is walking away from the camera is tracked with an 8-pixel block. This smaller block size is necessary to avoid multiple-motion being seen within the block caused by the other pedestrians.

Sequence	Frames Used	Description
CarParkB	[0, 105]	A car driving around the outside of a car park, taken from a security camera
Cyclist	[0, 100]	A cyclist moving slowly away down a road, taken from fixed camera
Pedfollow	[0, 175]	A pedestrian walking down a quiet street, taken from a moving camera
Snowseq	[20, 200]	Many pedestrians walking across a court–yard with heavy snow fall, taken from a fixed camera

Table 4.1: *Information on the general purpose test data sets used to test reference block updating strategies in Chapter 4.*

The results presented throughout this chapter are shown to 2 significant figures for clarity of presentation. However, it should be noted that due to the manual tracking process the ground truth is not entirely accurate. When comparing the accurate results it is not possible to compare accuracy greater than 1.5 pixels mean squared error. Furthermore, any accuracy comparison with figures less than 3 pixels mean squared error must be made with care. In general, a decimal change of less than 0.2 should be considered identical as no perceptible effect on the accuracy of the target track.

4.3 Investigation of Update Strategies

The aim for any strategy is to allow a genuine change in appearance to be represented in the reference block but to keep the reference block stable in the face of noise. It is possible to simply update the reference block after a preset number of frames but as Table 4.2 demonstrates,

it is difficult to chose an optimum update rate. In Table 4.2, the parameter T_s determines how many frames elapse before the reference region is replaced by the region of best match from the current image. Peacock *et al.* [124] investigated a number of alternative reference block updating strategies showing that the Kalman and FIR filter strategies demonstrate the highest accuracy when tracking with a region-based system. However, they do not discuss how to choose appropriate parameters for the filters; this can make a significant impact on the performance of the filters. These two filters are re-examined below and their accuracy and stability under varying parameters is investigated. Their robustness to image noise is an important aspect of their performance and this is also investigated.

Sequence	$T_s = \infty$	$T_s = 1$	$T_s = 5$	$T_s = 10$
CarParkB	4.2 (6.3)	LOST	4.1 (4.1)	1.9 (0.7)
Cyclist	2.1 (1.5)	37 (970)	3.1 (2.7)	2.2 (1.6)
Pedfollow	2.0 (6.0)	6.3 (12)	100 (3100)	91 (2900)
Snowseq	18 (110)	59 (1500)	25 (120)	37 (200)
Synth1	3.0 (3.0)	51 (1800)	1.9 (2.0)	2.0 (2.2)

Table 4.2: *Investigation into reference region updating, showing the mean error in pixels with the error variance in brackets (all numbers are to 2 s.f.). It is clear that to achieve optimal results the rate of update must be carefully selected. The accuracy for the synthetic Dervish simulator sequence (Synth1) is for translation only, although the rotation accuracy followed a similar trend.*

4.3.1 Error and Noise Models

The signal being filtered is the intensity (grey-level) of the reference block and any change within that. There are two types of error present in the data: image noise and motion estimation error. Image noise may be from the imaging device, transmission media or the digitisation process. It is reasonable to assume that in good lighting conditions the image noise is zero-mean Gaussian distributed [10, 126], which should be controlled by a correctly tuned filter. If more is known about the image sensor then it may be possible to characterise the noise more accurately. Certain types of image scene will also introduce image noise of a similar form; for example dust, rain and snow. Although this is not actually image noise, within the motion estimation context this is noise that will adversely effect the accuracy of the motion estimate and should be treated the same way as image noise. The second type of error, motion estimation error, will lead to incorrect position and this will shift all pixels away from their correct position.

The reference update filter will then attempt to filter an incorrect value into the data sequence. It is important that the filter also rejects motion estimation noise, otherwise the reference block may start to *drift* across the image embedding the motion estimation error within the reference block.

The motion estimation error is difficult to quantify as it is a logical error and cannot be estimated in the linear manner. One possibility would be to use a confidence measure on the motion estimation and this is discussed further in Section 4.5.2. In contrast, the image noise is relatively easy to quantify and Table 4.3 gives estimates for the image noise variance in each of the four test sequences. It is important that an online method is used for estimating the image noise variance to allow easy selection of optimum filter parameters. Therefore, the technique proposed by Rank *et al.* [127], which assumes that all noise is high-frequency and uses high-pass filtering to extract the noise was adopted. The Cyclist and Pedfollow sequences are both clean sequences and accordingly have low noise variances. The snow within Snowseq is regarded as noise and within our context this is correct; therefore, the snow sequence has a much higher image noise variance. The CarParkB sequence was captured with a low-quality CCTV camera and has very little image noise but does contain a considerable number of artefacts on the moving objects. This makes it a particularly suitable sequence for testing of the reference block updating algorithm.

Sequence	**Noise Variance** *(grey levels)*
CarParkB	1.2
Cyclist	2.4
Pedfollow	2.9
Snowseq	6.0

Table 4.3: *Estimates for the image noise variances, in grey levels, for each of the four test sequences used at this stage.*

Given the assumption of the correlation-based feature tracking, the reference block can be assumed to change slowly and smoothly. However, the sampling rate is relatively slow with a likely maximum of 25Hz. A person being tracked could turn their head or body and in less than one second completely change the target appearance. The average rate of change for the four test data sets was measured to be approximately 12 grey levels per frame, which is 5-percent of the total dynamic range (256 grey levels). Given a filter depth of 6, for example, the total

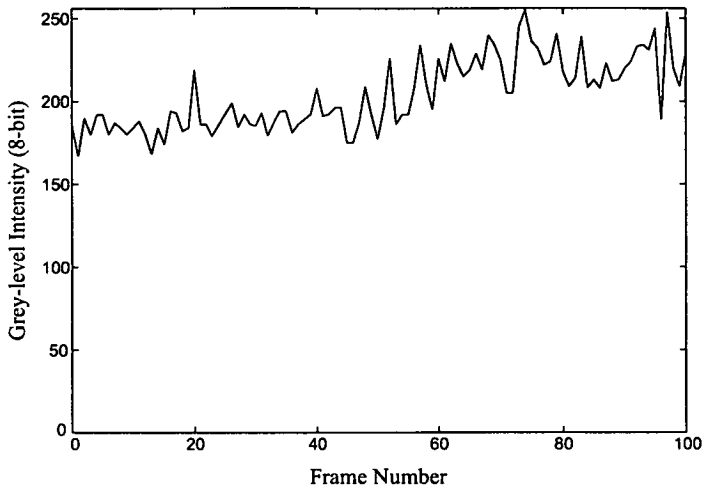


Figure 4.2: *Figure showing the intensity of a pixel in a moving block over 100 frames. The graph is scaled to the full range of a pixel (256 grey-levels). While the overall motion from start to finish is less than 1 grey level per frame, the motion between frames averaged approximately 12 grey levels.*

change possible within the filter is 72 grey levels or 30-percent. Figure 4.2 shows the pixel intensity value for the top-left corner pixel in each frame over the CarParkB test data set. It can be seen that much of the change is due to noise within the image. The overall intensity change on sequence CarParkB, shown in Figure 4.2, is 48 grey levels over 105 frames; approximately 1 grey level every two frames. The average acceleration is very small so a maximum acceleration of 1 grey level per frame is assumed.

4.3.2 FIR Filter

The FIR filter, reported in Peacock *et al.* [124], produced accurate and robust estimates on the selection of test sequences reported. However, in the general case, it is very expensive both in memory and computation. The FIR filter is described below in its intuitive non-recursive form for d non-zero coefficients a_i:

$$R_t = \sum_{i=1}^{d} a_i M_{t-i} \tag{4.1}$$

where R_t is the reference block at time t and M_t is the block at the position of best match at time t. The reference block is a linear combination of the previous d blocks of best match.

The parameters for the FIR filter are the depth (order) of the filter d and the coefficients a_i. There are many different design techniques for FIR filters but due to the nature of the data, which is inherently time-domain, the sophisticated frequency-domain techniques are not really applicable. The computational cost is $dn^2M + dn^2A$ where d is the depth of the filter, M is the cost of multiplication, A is the cost of an addition/subtraction and n^2 is the number of pixels in the reference region.

FIR Performance Optimization

Peacock *et al.* proposed for the FIR filter that $d = 5$ and $a_i = 0.9^{i-1}, a_0 = 1$, causing the weights to cascade down giving less significance to older data. In practice, this is a weighted version of the common moving average filter. A moving average filter is optimal at removing white-noise while keeping a sharp-step response [128]. In the time-domain, the filter is very good at smoothing but it performs badly in the frequency-domain when used as a low-pass filter. However, in its recursive form it is an extremely fast filter, requiring only one subtraction and one addition regardless of the depth of the filter. It is shown below in its recursive form, using the same notation as Equation 4.1:

$$R_{t+1} = M_t + R_t - M_{t-d} \tag{4.2}$$

Using the proposed FIR filter coefficients it is possible to rearrange the FIR filter into the form of a moving average filter (Equation 4.2) and as a result reduce the computational cost. The first stage of the rearrangement is to expand Equation 4.1, to give:

$$\begin{aligned} R_t &= \beta M_{t-1} + \beta^2 M_{t-2} + \ldots + \beta^{d-1} M_{t-d+1} + \beta^d M_{t-d} \\ R_{t+1} &= \beta M_t + \beta^2 M_{t-1} + \beta^3 M_{t-2} + \ldots + \beta^d M_{t-d+1} \end{aligned} \tag{4.3}$$

From this it can be seen that the coefficients are shifted by a factor β and the filter can be reorganised into a simpler recursive filter:

$$R_{t+1} = \beta \left(M_t + R_t - \beta^d M_{t-d} \right) \tag{4.4}$$

The reduction in computational cost is from a linear cost in terms of the filter depth to a constant cost. The overall cost is now $2n^2M + 3n^2A$ where M is the cost of multiplication, A is the cost of an addition/subtraction and n^2 is the number of pixels in the reference region. In comparison to the cost for a non-recursive FIR filter, this modification will be more efficient for a filter of depth 3 or greater. However, there is an associated reduction in flexibility of the filter but this does not appear to be significant. It may be possible to use the Blackman cascaded simple averaging filter [129] to further reduce the computational cost allowing for slight inaccuracies within the approximation. However, many of the assumptions made are now invalid; in particular, inclusion of a parallel multiplier/ALU on modern DSPs means algorithms that reduce the number of multiplications but increase the number of additions actually reduce the flexibility for implementation.

4.3.3 Kalman Filter

The Kalman filter was the most robust strategy reported by Peacock *et al.* [124] but a general Kalman filter per region tracked is very expensive. To address this issue, Peacock *et al.* noted that the observation space and state space are identical, leaving the transform H_n as simply the identity matrix. Furthermore, an underlying assumption for the BMA is that the appearance of the region tracked remains constant over time, so the state transition matrix Φ is also constant. If it is then assumed that the observation and dynamic model noise parameters are constant then the Kalman gain will settle down to a constant value, which could be pre-calculated. The operation of the Kalman filter then becomes a recursive filter of the form:

$$R_t = R_{t-1} + K(M_{t-1} - R_{t-1}) \tag{4.5}$$

An interesting area that was not explored in Peacock *et al.* [124] is when the region is undergoing a predictable transform that could be modelled in the state transition matrix. For example, a car surveillance camera mounted at the side of the road would track over a significant scale

change that would need to be accounted for as the vehicles move away, or towards, the camera.

4.3.4 Parameter Choice and Accuracy

After an initial investigation into the FIR filter it was determined that coefficient choice had a significant effect on the filter performance. The higher the filter coefficients the longer a result will have a significant effect on the output of the filter, therefore, the deeper the filter can be. However, filter depths of only 10 or less were considered because the memory storage requirements become too great. With the constraint on the coefficients for the optimized FIR filter of $a_i = \beta^i, i = 1 \ldots O$, values between 0.75 and 0.95 were tested for β. The results are shown in Table 4.4 and it can be seen that the most accurate situation is when β approaches unity with a longer filter length. However, because of the extra storage cost the practical use of this filter is limited.

		FIR Coefficient (β)				
		0.75	**0.80**	**0.85**	**0.90**	**0.95**
FIR Depth	**4**	2.3 (1.3)	4.8 (9.5)	4.8 (9.5)	4.0 (5.1)	13 (180)
	5	2.0 (1.2)	4.4 (6.0)	14 (210)	2.2 (1.6)	2.7 (2.2)
	6	2.1 (1.2)	2.1 (1.3)	2.1 (1.3)	2.2 (1.3)	2.2 (1.3)
	7	2.1 (1.4)	2.0 (1.2)	2.3 (1.2)	1.8 (1.3)	1.8 (1.3)
	8	5.5 (12)	2.1 (1.4)	2.0 (1.4)	1.7 (1.3)	1.9 (1.4)
	10	LOST	2.0 (1.6)	1.8 (1.6)	1.8 (1.6)	2.0 (1.6)

Table 4.4: *Results for FIR filter update strategy on the CarParkB data set showing the mean error (in pixels), with error variance in brackets, for the position estimates. LOST indicates that the target being tracked was lost.*

There are two parameters for the Kalman filter: the dynamic model noise covariance matrix and the observation error covariance matrix. However, region-based correlation tracking assumes pixels are spatially-rigid but otherwise independent so only the variance and not covariance need be found. It is the relative magnitudes and not the absolute values that effect performance. In Table 4.5 it is shown that a balance of 50:1 between dynamic model noise and observation noise proved the most accurate on all the test sequences.

It is possible that small improvements in accuracy can be made by adjusting the balance depending on the sequence. A possible method for selecting parameter values is given below. Starting with the assumption that regions transform rigidly from one frame to the next, no large

Observation Noise	Dynamic Model Noise 0.001	0.01	0.1	1	10
0.05	1.6 (1.1)	3.3 (4.0)	20 (260)	LOST	LOST
0.5	9.1 (16)	1.6 (1.1)	3.3 (4.0)	20 (260)	LOST
5	9.6 (22)	10 (21)	1.6 (1.1)	3.3 (4.0)	20 (260)
50	11 (32)	11 (31)	18 (160)	1.6 (1.3)	3.3 (4.0)
500	10 (40)	10 (39)	11 (31)	18 (150)	1.9 (1.3)

Table 4.5: *Results for the Kalman filter update strategy on the CarParkB data set showing the mean error (in pixels), with the error variance in brackets, for the position estimates. LOST indicates that the target being tracked was lost.*

intensity changes are anticipated from frame to frame so the observation noise variance σ^2_{obs} can be set to 100. Assuming a normal distribution for model noise would give 95% of pixel intensity changes within 20 intensity levels. This value is a balance between the slow change assumed by the block matching and the low sampling rate of the imaging device. Using the ratio of 50:1, the dynamic model noise should then be approximately 2. This is relatively close to the estimate of the image noise variance for each of the sequences, which averaged 3, and image noise variance could be used to adjust the dynamic model noise.

The Kalman filter is being used to calculate the weight for a recursive filter under a constant operation. However, if the situation were unknown or were to change the previously optimized filter performance would degrade. One area where the parameters will change is in track-initiation. A Kalman filter is self-starting, which can be seen by the changing Kalman gain (Figure 4.3), but the recursive implementation is not. Table 4.6 shows the four test sequences tracked with a complete Kalman filter and with the optimized version. It can be seen that the optimized version generally performs worse, although this depends on the specific situation.

4.3.5 Accuracy in the Presence of Noise

The accuracy reported for both Kalman and FIR update strategies is very good but the actual data sets have low levels of image noise present; see Table 4.3. However, in many real world applications the levels of image noise will be considerably higher and it is important that the reference update strategies continue to provide increased accuracy. Therefore, the performance of the two update strategies under artificial Gaussian-distributed additive noise on the image

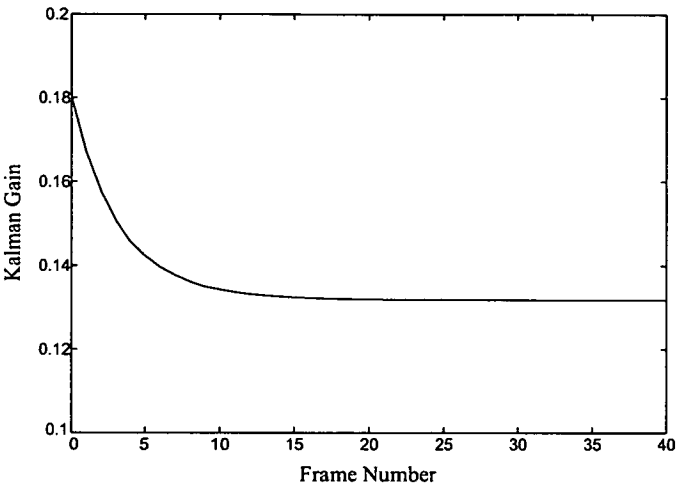


Figure 4.3: *Figure showing the Kalman gain over the first forty frames as it settles down to a constant value of approximately 0.137.*

Sequence	Reference Update Filter		
	Single	**Optimized**	**Kalman**
CarParkB	11 (30)	1.6 (1.1)	18 (150)
Cyclist	2.1 (1.5)	3.1 (2.8)	3.1 (2.8)
Pedfollow	2.0 (6.0)	1.5 (0.5)	1.7 (0.6)
Snowseq	18 (110)	16 (69)	21 (190)

Table 4.6: *The performance of a complete Kalman filter is compared with the performance of the optimized (recursive) Kalman filter. The complete Kalman filter performs significantly better, mainly due to poor track-initialisation in the optimized version.*

sequences was investigated. The variance for the artificial noise is defined as σ_N. The range of noise added is intended to cover a wide range of conditions in which region-based tracking may have to operate, although noise reduction techniques would normally be applied to data with high levels of noise ($\sigma_N > 15$) prior to block matching.

Figure 4.7 shows the accuracy of the Kalman filter and FIR filter update strategies applied to the data sets. All data sets contain significant changes in target appearance over the image sequence but do not contain occlusion of the target; both strategies show graceful degradation in accuracy as the level of image noise increases. The Snowseq is noisy already and, as such, very hard to track. It can be seen from the results that the addition of artificial noise produces a few odd results and that care must be taken when applying the reference block update strategies to image sequences with high levels of image noise.

Sequence	Filter	Standard Deviation (σ_n) for Artificial Image Noise				
		None	1	5	15	30
CarParkB	None	11.4 (30.4)	7.7 (31.7)	5.1 (22.0)	LOST	LOST
	FIR	2.1 (1.4)	2.1 (1.4)	2.1 (1.4)	LOST	LOST
	Kalman	1.6 (1.1)	LOST	LOST	LOST	LOST
Cyclist	None	1.4 (1.5)	1.4 (1.5)	1.4 (1.5)	7.5 (140)	3.1 (16)
	FIR	1.4 (0.5)	1.4 (0.5)	1.2 (0.3)	1.1 (0.4)	4.0 (3.6)
	Kalman	1.1 (0.3)	1.1 (0.3)	1.1 (0.3)	1.4 (0.6)	2.1 (1.1)
Pedfollow	None	2.0 (6.0)	3.6 (46)	3.5 (44)	3.7 (47)	5.4 (42)
	FIR	1.3 (0.3)	1.3 (0.3)	1.3 (0.3)	1.8 (0.5)	1.8 (0.6)
	Kalman	1.5 (0.5)	1.5 (0.5)	1.6 (0.5)	1.6 (0.7)	LOST
Snowseq	None	18 (110)	18 (120)	18 (110)	12 (40)	LOST
	FIR	20 (110)	21 (110)	20 (73)	LOST	LOST
	Kalman	16 (69)	16 (69)	24 (204)	LOST	LOST

Table 4.7: *The four general purpose data sets tested with three different update strategies (None, FIR, Kalman) under varying amounts of artificial Gaussian-distributed noise. It can be seen that the FIR and Kalman update strategies generally provide good performance under noise.*

Using the previous strategy to vary the observation noise variance, the Kalman filter was re-tested with two different values for Observation noise. Table 4.8 shows the results. Some performance improvement can be seen but the ratio is very sensitive and too much change can dramatically reduce accuracy.

4.3.6 Summary

The FIR filter provides good performance with the optimizations presented, giving both excellent accuracy and relatively low computational cost. However, as with all fixed-memory filters the memory cost is dependent on the depth of the filter d, forcing the use of small filter depths to avoid large memory overheads. Given that every feature tracked would require a separate filter the memory cost is likely to be too great for most applications, especially a DSP-based real-time application where high-speed memory is limited.

The Kalman filter provided the greatest accuracy and under optimization also produced the most efficient recursive implementation, only requiring one previous block to be stored in memory. However, the optimized version does not produce as accurate results, probably due to the lack

Sequence	Kalman Observation Noise	Standard Deviation (σ_n) for Artificial Image Noise				
		None	1	5	15	30
CarParkB	3	1.6 (1.1)	2.3 (1.3)	2.5 (1.4)	LOST	LOST
	5	1.6 (1.1)	LOST	LOST	LOST	LOST
Cyclist	3	1.1 (0.3)	1.1 (0.3)	1.1 (0.3)	1.4 (0.6)	2.1 (1.1)
	5	1.1 (0.3)	1.1 (0.3)	1.1 (0.3)	1.4 (0.6)	2.1 (1.1)
Pedfollow	3	1.4 (0.4)	1.6 (0.4)	1.5 (0.4)	24 (1200)	LOST
	5	1.5 (0.5)	1.5 (0.5)	1.6 (0.5)	1.6 (0.7)	LOST
Snowseq	3	21 (310)	24 (340)	31 (400)	33 (390)	LOST
	5	16 (69)	16 (69)	24 (204)	LOST	LOST

Table 4.8: *The four general purpose data sets tested with varying values for the observation noise variance under varying amounts of artificial Gaussian-distributed noise. It can be seen that some performance improvement can be achieved.*

of track-initiation. Furthermore, the selection of model noise variance and observation noise variance values are subject to a difficult selection process. If a dynamic selection system was employed the accuracy of the tracking may be increased but, in this case, the optimization could not be performed and a complete Kalman filter would be necessary per feature tracked.

The next section proposes a new update strategy based on a recursive least-square polynomial filter. In continuing operation the recursive filter is identical to the optimized Kalman filter (Equation 4.5). However, the parameter choice is significantly easier and less subjective. Furthermore, correct track-initiation is performed increasing the accuracy of the optimized filter at no extra computational cost.

4.4 Least-Squares Reference Block Updating

4.4.1 Discounted Least-Squares

The FIR filter provides very robust performance and taking the optimized FIR filter shown in Equation 4.4 it can be seen that this is a discounted fixed-memory filter. However, fixed-memory filters have a high cost in both computation and memory. The FIR performance optimization partially addressed the cost of computation but still required d previous blocks to be stored. Instead it is possible to use a discounted least-squares (fading-memory) filter to obtain similar robustness but with lower memory costs. Brookner [117] provides a detailed derivation

of this filter, but the key difference is that the optimized FIR filter has a hard cut-off for old data while the a discounted least-squares (DLS) filter fades away indefinitely (semi-infinite). This suggests that a DLS filter could provide most of the benefits of the FIR filter and optimized Kalman filter. Furthermore, the DLS filter allows a more structured approach to filter selection and parameter choice.

The DLS filter is a polynomial fitting filter and normally the orthogonal Laguerre polynomial is employed, represented as $p^* = [p^*(r)]_t$. The data vector is defined as y_t with the discounting scheme of $\theta^r, 0 < \theta < 1$. Therefore, the expression to be minimized is:

$$e_t = \sum_{r=0}^{\infty} (M_{t-r} - [p^*(r)]_t) \tag{4.6}$$

It is considered that filter output R_t is an estimate for data vector y_{t+1} based upon the data available at time t. The update of the filter is based upon the error between the estimate and the actual data, defined as $\epsilon_t = (y_t - R_{t-1})$. The resulting steady-state filter is:

$$R_t = R_{t-1} + (1 - \theta)\, \epsilon_t \tag{4.7}$$

In this form the filter has an overall computational cost of $n^2M + 2n^2A$ where M is the cost of multiplication, A is the cost of addition/subtraction and n^2 is the number of pixels. This is more efficient than the optimized FIR filter. Furthermore, the filter only requires the previous reference block so memory usage is also very efficient in comparison to the optimized FIR filter which requires d previous blocks to be stored.

At this stage it can be noted that the optimized Kalman filter shown in Equation 4.5 is actually a steady-state discounted least-squares filter. However, the Kalman filter implementation required two arbitrary parameters to be chosen and balanced before the Kalman filter is run over an extended period to produce the Kalman gain. Instead, use of the DLS filter allows the update parameter θ to be chosen directly and suggests a suitable track-initiation procedure, both of which are detailed in the following subsections.

4.4.2 DLS Parameter Choice

For efficiency only a steady-state filter is considered with target velocity and acceleration ignored. Therefore, the only parameter within the DLS is θ and, using the design technique for a critically damped filter given in Chapter 1 of Brookner [117], the three parameters we need for the design are the standard deviation for the random error (σ_x), the maximum acceleration for the input signal ($\ddot{x}$) and the desired standard deviation for the filter prediction error ($\sigma_{t+1,t}$). However, instead of identifying the desired standard deviation for filter prediction error, we can define it in terms of sampling rate, which is predetermined. From Brookner [117], the systematic error is given by:

$$b^* = 3\sigma_{t+1,t} = \frac{\ddot{x}T^2}{(1-\theta)^2} \tag{4.8}$$

where $1/T$ is the sampling frequency. From Brookner [117], the normalised variance, or variance reduction factor, for a one-step predictor is given by both:

$$\begin{aligned} VRF_{DLS} &= \frac{(1-\theta)}{(1+\theta)} \\ VRF_{DLS} &= \frac{\sigma_{t+1,t}}{\sigma_x^2} \end{aligned}$$

Rearranging Equation 4.8 to give $\sigma_{t+1,t}$ it is possible to equate the two forms of the VRF_{DLS}, leaving θ as the only unknown. This produces:

$$\begin{aligned} \frac{(1-\theta)}{(1+\theta)} &= \frac{\ddot{x}T^2}{3(1-\theta)^2} \times \frac{1}{\sigma_x^2} \\ \frac{(1-\theta)^3}{(1+\theta)} &= \frac{\ddot{x}T^2}{3\sigma_x^2} \end{aligned} \tag{4.9}$$

Using the noise analysis in Section 4.2 to identify parameters, the maximum acceleration is set to 1 grey-level per sec per sec. The image noise variance estimates for the four general-purpose data sets have an average of about 3 (grey levels), so σ_x is set to 3. Given a sampling rate of

25 frames per second, $T = 0.04$. Therefore, the solution in this case is $\theta = 0.95$. It should be noted that this method is a simplification and a more rigorous presentation is given in Brookner [117]. However, the technique is simple and effective for the application and utilises easily accessible and intuitive design parameters.

An alternative technique would be to determine the equivalent length of the fixed-memory (FIR filter) required; Morrison (page 535) [130] provides further information on choice of θ in this case. However, using the calculation for a steady-state filter using the position variance, which is given below in Equation 4.10, the equivalent filter length for $\theta = 0.95$ is approximately 40.

$$L = \frac{2.00}{1 - \theta} \tag{4.10}$$

4.4.3 Track Initiation

The Kalman filter is a self-starting filter and it was demonstrated in the previous section that the Kalman filter is more effective when it has an initial period of rapid adjustment before settling down into a steady state (Figure 4.3 and Table 4.6). The DLS filter is not a self-starting filter and correct track initiation still presents a problem. However, the expanding-memory polynomial (EMP) filter, which is a close relative of the DLS, provides a good way of replicating the Kalman gain behaviour. This filter is self-starting and can be implemented in a very efficient recursive form, with the resulting steady-state filter shown below in Equation 4.11. This result is from Brookner [117] which provides a more detailed derivation and discussion for this filter.

$$R_t = R_{t-1} + \frac{1}{t+1}\epsilon_n \tag{4.11}$$

In practice, this is very similar to the steady-state form of the DLS but in this case the update weight is not based on θ but on the number of samples n. The filter starts off with a higher value, allowing more rapid adjustment, which slowly tails off after the initial starting period. However, the expanding memory filter cannot be utilised indefinitely as the systematic errors continue to increase with each iteration. The stage at which the change should occur is when the expanding memory filter has the same variance as the DLS filter. At this time the two filters

have the same memory and their systematic errors will approximately match. Again using the normalised variance for the filter, called the variance reduction factor (VRF), for each filter the balance point is:

$$\begin{aligned} VRF_{DLS} &= VRF_{EMP} \\ \left(\frac{1-\theta}{1+\theta}\right) &= \left(\frac{1}{t+1}\right) \end{aligned} \tag{4.12}$$

In the case of $\theta = 0.95$ the balance point, using the above equation, is $t = 38$. Therefore, the expanding-memory filter should be employed (Equation 4.11) until time step (t) is equal to 38. From then on the discounted least-squares filter should be employed using the memory from the expanding-memory filter. There is no increase in computational cost because the filter memory is transferred from one filter to the next and this track initiation can significantly improve performance.

Figure 4.4 (a) shows a pixel from the tracked block in CarParkB tracked with the DLS filter and with the track-initiated DLS filter. It can be seen that the track-initiation allows much more rapid adjustment over the first few frames, in case track-initiation is poor, but as the switch occurs ($t = 37$) their is very little difference in numerical value of the filters. In (b) the track is started one frame later, at a poor initial position, and the track-initiation responds very quickly to settle at a good position while the DLS filter lags well behind the true value.

4.4.4 Accuracy

The accuracy of the new track-initiated DLS reference block updating filter was tested on the four general-purpose data sets. The EMP filter was employed until $t = 37$ and from then onwards the DLS was used with $\theta = 0.95$. The results are shown in Table 4.9, the Kalman filter results are shown for comparison.

It can be seen that the track-initiated DLS based filter achieves performance roughly equivalent to the full Kalman filter with the computational cost of the optimized Kalman filter. This suggests that the proposed track-initiated DLS filter provides a good trade-off between accuracy and performance, combined with a sound theoretical procedure for track-initiation and parameter selection. The accuracy of the DLS with noisy images is almost identical to the Kalman

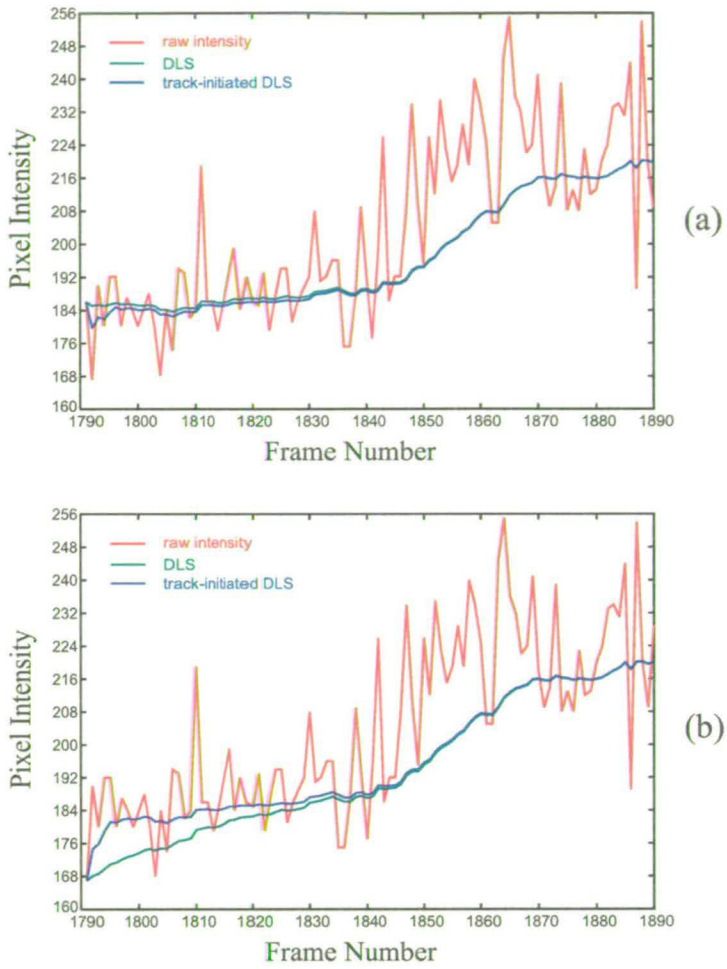


Figure 4.4: *Figure showing the intensity of a pixel in a moving block over 100 frames in the CarParkB sequence. The full range of a pixel is 256 grey-levels. The DLS filter and track-initiated DLS filter are shown, with the rapid adjustment of the track-initiation clearly visible. In (a) the initial track estimate is good but in (b) the initial estimate is fairly poor.*

filter

4.5 Further Discussion

There are a number of issues remaining within all the reference block update strategies when applying them to different situations. Sub-pixel motion, motion estimation error and handling of complex motion are all discussed in this section and possible solutions presented, although the best solution is often dependent on the application. It should be noted that sub-pixel motion and motion estimation error are, to some extent, related but have been broken into two

Sequence	Reference Update Filter		
	Kalman	Optimized Kalman	track–initiated DLS
CarParkB	1.6 (1.1)	18 (150)	1.4 (0.4)
Cyclist	3.1 (2.8)	3.1 (2.8)	1.8 (0.9)
Pedfollow	1.5 (0.5)	1.7 (0.6)	1.7 (1.6)
Snowseq	16 (69)	21 (190)	18 (86)

Table 4.9: *The four general purpose data sets are tested using the Kalman filter, in both full and optimized form, and the track-initiated DLS filter. It can be seen that the track-initiated DLS filter achieves similar accuracy as the full Kalman filter but at a computational cost close to the optimized Kalman filter.*

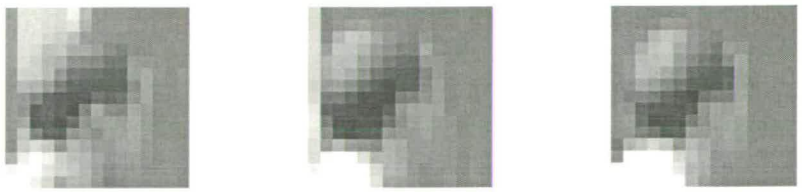
subsections for ease of explanation.

4.5.1 Sub-pixel Motion

Through the digitisation process the world is broken down into pixels but the real-world does not operate on a pixel basis, resulting in objects actually moving continuously. However, the update filters are fed with only pixel accurate information giving alignment errors within the data. These minor alignment errors could, over time, blur the reference image. Furthermore, the sub-pixel inaccuracies could create a shift within the reference block of, say, $\frac{1}{2}$ pixel. If this were repeated it could gradually move the reference block away from the true location.

Taking the two possible scenarios to extreme, it is easy to demonstrate that oscillation between two directions can blur the reference block and that a slow movement in a single direction can force the reference block away from the original target. It is easy to demonstrate the effect of both these situations on the filter with degenerate artificial sequences. While neither of these situations is likely in extreme, they can both effect the accuracy of the reference block.

In testing, oscillation was not identified as a major problem and excessive blur was not found, as shown in Figure 4.5. However, if it is known that oscillation may occur in an application it may be necessary to take steps to avoid it. Reference block drifting, on the other hand, did prove that it could be a problem when there was no significant movement in the image. To avoid this problem it is possible to slow the updating of the filter when the motion detected is below a certain threshold for an extended period. However, from an vehicle motion estimation perspective it may be helpful to reset the entire feature tracking system on a regular basis when



frame 001 *frame 100* *frame 174*

Figure 4.5: *The reference block, filtered with the track-initiated DLS update strategy, is shown at three different stages of the sequence. It can be seen that no blurring occurs and the image remains sharp throughout. The background pixels (vertical strip far right), which continuously change balance into a medium grey.*

there is no motion, in an effort to avoid this problem.

4.5.2 Confidence Measure

Motion estimation error, which was introduced in Section 4.2, is a logical error and while the filter will, to some extent, naturally tackle this problem the motion estimation error can propagate through the filter and effect further motion estimation. The only way to tackle this is through a confidence measure on the motion estimate. This way the reference block update filter will know if the new contribution is reliable and can discard or weight it accordingly. It is important to first determine if an accurate confidence measure will allow an improvement in the reference block updating. Therefore, the four test sequences were tested when the ground truth was known, giving a 100-percent accurate confidence measure. In this case the correlation between the errors and the confidence measure is 1. The artificial confidence measure was also tested with slight errors, reducing the overall correlation to 0.8 and 0.9. In the tests, if any mean squared error exceeded 5 pixels, the data was discarded as too inaccurate, otherwise it was weighted accordingly. All correlation was calculated using Pearson's R [30]. The results are shown in Table 4.10.

It can be seen from the results in Table 4.10 that an accurate confidence measure can increase the accuracy of the overall track but not significantly. In the case of the less accurate confidence measure there was no performance gain and, in fact, it may have a negative impact.

Confidence measures for region correspondence were discussed in Chapter 3 and a number of these have been investigated for correlation-based motion estimation. However, because the final application is for passive navigation the confidence measures with high computational

Sequence	Confidence Measure			
	None	0.8	0.9	1.0
CarParkB	1.6 (1.1)	2.1 (1.7)	1.8 (1.3)	1.3 (1.0)
Cyclist	3.1 (2.8)	3.2 (3.2)	3.0 (2.8)	3.0 (2.7)
Pedfollow	1.5 (0.5)	1.5 (0.5)	1.3 (0.7)	1.4 (0.5)
Snowseq	16 (69)	16 (71)	17 (72)	15 (54)

Table 4.10: *Results for the accuracy of the motion estimation achieved given an artificial confidence measure of varying accuracy. The confidence measure must be very accurate (>0.9 correlation) to improve the accuracy of the motion estimation.*

cost have not been investigated. The first of three confidence measures tested is a threshold on the the value of the correlation measure using both SAD and ZNCC. This is a cost-free confidence measure allowing easy use in passive navigation. The second confidence measure calculates a rolling mean for the similarity measure and determines how far the current minimum SAD/ZNCC is from the mean. The final confidence measure was proposed by Singh and Allen [19] and is based on the covariance matrix for their least-square position estimate. The technique is more complex, although it is by no means the most complex technique available and is included for comparison. The results are shown in Table 4.11.

Sequence	*min*(SAD)	*deviation*(SAD)	Singh
CarParkB	0.58	0.45	0.55 / 0.57
Cyclist	0.62	0.56	0.31 / 0.32
Pedfollow	0.61	0.50	0.86 / 0.89
Snowseq	0.34	0.36	0.22 / 0.28

Table 4.11: *The four general-purpose data sets are tested using three different confidence measures. It can be seen that no confidence measure performs particularly well, with the $min(SAD)$ proving most consistent.*

It can be seen from these results that the confidence measures are not consistently accurate enough to provide useful information for the reference block update filter. In fact, none of the confidence measures proved particularly accurate. Furthermore, discarding data complicates the filter because of the irregular time-spacing of the data. Although filtering techniques exist for irregular timed data this adds further computational cost to the filter while providing questionable benefit.

4.5.3 Complex Motion Models

Correlation-based feature correspondence has traditionally used translation motion models. However, more complex motion models have recently been defined to allow for greater accuracy in motion estimation. For example, the affine model can be used to detect arbitrary motion of a block at an increase in computational cost. To correctly deal with the complex motion the newly acquired block at the position of best match must be *untransformed* before being applied to the update filter.

However, the reference block will have to be *transformed* before being used in the correlation search and *untransformed* based on the position of best match, leading to further opportunities for error within the pixel position and intensity. Furthermore, the mapping between transformations can lead to complex spatial interpolation problems, such as rotation about an axis parallel to the viewing window or a significant scale reduction. For these reasons, use of reference block updating with complex motion models would need to be investigated on a case-by-case basis.

4.6 Summary

In this chapter, the FIR and Kalman reference block update strategies proposed by Peacock *et al.* [124] were investigated as they demonstrated significant accuracy improvements when tracking with correlation-based systems. The FIR filter was reorganised into a recursive form, with some loss of flexibility, to reduce the computational cost but no reduction was possible in the storage cost. Both update strategies proved accurate and robust to noise. The Kalman filter proved the most attractive due to the low computational and storage costs. However, the choice of two noise variances is highly subjective and they must then be placed into a Kalman filter to get a constant Kalman gain.

The track-initiated DLS filter was proposed to replace the Kalman filter. In its optimized recursive form it is identical, therefore, providing equivalent accuracy and performance but the track-initiated DLS filter allows procedural selection of only a single parameter θ. Furthermore, this parameter is used directly in the filter and any changes to it are clearly transmitted through to the filter. This is in contrast to the Kalman filter, where the changes to the parameters have an indirect effect on the output.

Track initiation was also proposed for the track-initiated DLS filter through use of an expanding-

memory polynomial filter. Using a region-based tracker it is assumed that the reference block remains constant throughout the track and it could be argued that track initiation is not required because the original estimate is in fact correct. However, using reference block updating a constant reference block is no longer assumed. Furthermore, except for special situations where *a priori* information is available about the target it is not possible to guarantee that the initial reference block is a good representation of the target. In this case, it is necessary to use track initiation and the expanding-memory polynomial filter is a good method [117] for the proposed DLS filter.

The issue of motion estimation error was addressed through the use of a confidence measure of the accuracy of the estimate. However, it was demonstrated that the confidence measure would have to achieve a very high accuracy, with a correlation > 0.9, and that none of the simple confidence measures tested achieve a high enough accuracy. It is possible that a more complex confidence measure could be employed if the application allowed the extra computational cost.

This chapter concludes that the DLS filter with EMP track initiation can provide a significant increase in accuracy when using correlation-based feature tracking. The performance remains good when using optimized recursive filters and is of the order $n^2M + 2n^2A$. Within a target tracking application, after reviewing the accuracy / performance trade-off, the increased accuracy from a reference block updating strategy may be an advantage. However, for estimation of vehicle motion, where larger numbers of features are tracked for varying lengths of time it is not clear if any advantage is gained; this is addressed further in Chapter 7.

Chapter 5
Hardware Implementation for the Dervish landmine-clearance vehicle

5.1 Introduction

Chapter 2 introduced the Dervish landmine-clearance vehicle and some of the application constraints it placed on the system. Chapter 3 provided a review of motion estimation algorithms and selected an appropriate range of algorithms. The final constraint on the system is, therefore, the hardware. Section 5.2 starts with a brief overview on how the algorithm constraints effect the hardware choice. Section 5.3 then goes on to summarise possible hardware platforms for the project, before deciding on the the most appropriate, given all the various requirements. Section 5.4 discusses the implementation issues with the specific hardware platform chosen, while Section 5.5 goes on to detail some DSP design issues with the hardware platform.

5.2 Image Processing Overview

Chapter 3 provided a summary of motion estimation algorithms and in conclusion correlation-based feature tracking was selected as the most appropriate choice. Correlation-based feature tracking algorithms vary significantly and a detailed analysis would not be of benefit. However, the algorithms all share some common details that can affect choice of hardware platform.

The use of a region, normally a square block, provides a local area over which a single calculation must be repeated. The exact nature of this calculation will vary but normally a calculation is between two distinct and small areas. This local computation is then repeated at every feature throughout the image. Most of the decision making process is located at a higher level of the algorithm and not in the high-cost local computation stage.

The repetition and summation of a simple calculation, for example the squared difference, over a local area can be adapted to make good use of any multiply-and-accumulate routines within the hardware. The calculation can also be performed in parallel or through use of a low-latency

loop. A large amount of local memory, such as cache or SRAM, will provide a significant advantage to the calculation. Ideally, the entire feature matching procedure for a single feature would be performed without access to the slower, bulk memory (DRAM). Obviously, a significant amount of bulk memory would be helpful, especially during development where extensive test and debug information may be generated. The input video signal will have to be transferred to bulk memory and the video data will then have pre-processing before being passed on to the feature tracking algorithm. A separate I/O processor that can handle the input video signal without affecting the rest of the system will provide an increase in overall performance.

In contrast, differential-based optical flow performs a number of calculations across the entire image before entering a decision making process. This procedure is normally iterated a number of times. It is, therefore, necessary to have access to two full images making it difficult to avoid regular access to slower bulk-memory. In general, any decision on a suitable hardware platform for a different motion estimation algorithm would require a new analysis and may well arrive at a different decision.

5.3 Choice of Hardware Platform

5.3.1 Overview

The choice of hardware platform is a moving target and this review was undertaken between October 1999 and January 2000. A review now may well reach a different set of conclusions. There are many different options for a suitable hardware platform but for this review they have been divided into four categories: custom silicon, high-performance FPGA, specialist image processing platform and general-purpose DSP. Each of these categories is briefly discussed in turn before the most appropriate platform for this application is chosen. A single general-purpose processor, such as an Intel Pentium, was not sufficiently powerful when the decision was made.

5.3.2 Custom Silicon

A custom hardware solution could potentially provide an excellent combination of high performance, low power consumption and low physical space. In particular, using custom silicon to provide image pyramid, correlation and warping units would be appropriate and such units

have been used in applications where high-performance is necessary [122]. Other, reconfigurable, VLSI architectures have also been developed for more general use in motion estimation [131].

However, the complexity and turn-around time of the design task required for a custom ASIC and the cost of production when the volume required is very low does not make this option cost-effective. Furthermore, the ASIC would need to be integrated into a suitable platform for inclusion onto the Dervish, which significantly increases the complexity of the engineering task. The scale of the development task for a custom ASIC would normally be in many tens of man months with a minimum prototype cost of one hundred-thousand pounds. For these reasons a custom hardware solution is not considered suitable for the Dervish landmine-clearance vehicle.

5.3.3 High-Performance FPGA

Current High-Performance FPGA solutions, such as the Virtex II from Xilinx [132], provide much of the flexibility of custom ASIC solutions but offer a more automated design flow and much faster development cycle time. Instead of tens of man months, a FPGA design would normally take closer to 5 man months. DSP Cores are widely available for many FPGA architectures [133] allowing rapid implementation of common algorithms. Many of the functional units from a custom silicon design could be implemented on a FPGA at a similar level of performance. A recent assessment of FPGA suitability for real-time motion estimation concluded that a modern FPGA was a suitable platform for real-time correlation-based motion estimation [134].

The major drawback to a FPGA approach is the considerable development cost required to integrate the FPGA into a complete system. Most FPGA platforms currently combine a general-purpose processor, such as a ARM RISC processor [135] and I/O controllers on a PCB. There is also an increase in development time required for algorithm implementation on the target FPGA's platform.

A new generation of Platform FPGA solutions is attempting to address the issue of system integration by adding extended system I/O features and dedicated optimized circuitry, such as high performance arithmetic and dual-ported RAM, to the standard FPGA core modules [132]. However, these are a very recent and developing technology and were not ready for immediate integration in January 2000.

5.3.4 Specialist Image Processing Platforms

Specialist image processing platforms have been designed around the needs of current image processing algorithms and implement development libraries and an environment to allow rapid use. Over the past 20 years these have been used in many vision tasks, such as the DARPA funded ALV [136], which used the VICOM image processor and three general purpose processors. The Philip's Tri-Media [137] image processor, Coreco Advanced Pixel Processor [138] and Datacube [139] are current examples of specialist image processing platforms.

Normally specialist image processing platforms provide a series of individual computation units connected by high speed buses. The flexibility and programmability of these units varies significantly between platforms. If an image processing algorithm maps efficiently onto the target platform the performance increase can be one order of magnitude, if not more. However, if the image processing algorithm does not map efficiently the overall performance can be very poor. Two possible ways that the algorithm may not map efficiently are that image representation does not fit into individual computation units, for example the colour bit-depth or spatial resolution, or that the organisation of the computation units and buses does not fit the algorithm.

5.3.5 Digital Signal Processors

The current generation of high-performance general-purpose Digital Signal Processors (DSP) provide peak performance of around 1 Gigaflops per chip [140], utilising multiple computation blocks (SIMD) with each computational block being capable of multiple operations per cycle (VLIW). This complex structure can produce very high-levels of performance but it can also be difficult to map algorithms onto the DSP, with poorly mapped algorithms suffering performance degradation in the order of ten-times (e.g. possibly only 100 Megaflops).

A general-purpose DSP is unlikely to ever be able to compete on a direct chip-to-chip basis with the flexibility and parallelism of a good custom silicon or FPGA implementation. However, the current generation of general-purpose DSPs are reaching the 1 Gigaflops (peak performance) barrier and are available in eight DSP, or even sixteen DSP, platforms. This will increase the total area and power consumption and while this may be significant for some applications it should not present a problem for the Dervish application. One area that will effect the Dervish is the exponential price curve for modern DSP chips, where a single new generation chip may cost ten times more than the previous generation chips but only perform two or three times

better.

Multiple DSPs increase design complexity but modern development tools reduce the impact and, as already stated, image processing algorithms are relatively easy to split either by data (sub-divide image) or by functionality (e.g. pre-processing, motion estimation, estimation filtering) leading to an obvious distribution of load.

General-purpose DSPs are normally integrated into a final system with general-purpose processors and I/O controllers, probably on a custom PCB to reduce the redundancy and area of the final system. However, DSP boards are available that can run independent of any further general-purpose processors [141, 142] and they contain integrated I/O solutions, such as RS232, Ethernet and SHARCPAC [143]. These allow very rapid integration of the DSP with all external systems, such as a frame-grabber or electronic compass.

5.3.6 Analysis

Specialist image processing platforms provide an interesting alternative but by their very nature, require highly specialist knowledge that is not transferable. Furthermore, the relatively high cost compared with the excellent performance of modern FPGA and DSP chips makes them an unsuitable option. The major advantage of a custom silicon or FPGA design is that image processing algorithms can be broken into pixel-level calculations based on the specific algorithm, allowing a high degree of parallelisation. However, given the short development time and low production volume, custom silicon is not a practical choice.

Therefore, it was concluded that there are only two viable alternatives: high-performance FPGAs and general-purpose DSPs. In an ideal situation, where suitable resources are available for development, a FPGA solution might provide an attractive solution. Alternatively, a FPGA-DSP combined solution may provide a lower cost development with only minor reduction in performance. Bittware [142] have recently released a DSP board with an on-board Virtex II FPGA; Altera alternatives are also available. Alternatively, Xilinx [132] have just announced their Virtex II Pro FPGA that has an on-chip PowerPC. However, these are all at too early a stage of release for integration into this system but would provide interesting alternatives should the hardware platform be reconsidered.

One final and recent development is that many DSP boards are offering Pentium alternatives [138], claiming a better price-performance ratio with modern 500MHz+ Pentium III chips.

There are associated disadvantages, such as higher power consumption, but the readily available operating systems (including true real-time) and software development tools makes the Pentium boards an attractive alternative to the complex, and often unreliable, software associated with DSP chips. Unfortunately, this development came too late to be considered for this system.

The decision is based on what DSP development boards were available between January 2000 and November 2000 when the purchases were made. At that stage the two main high-performance general-purpose DSP platforms were the Texas Instruments C'60 series and the Analog Devices SHARC series. The two platforms have similar architectures, with the key difference being the greater memory of the SHARC boards versus some extra computation on the C'60. After consideration of reviews and comparisons [140, 144] it was decided that neither platform offered any obvious advantage. The Analog Devices SHARC series was chosen because of extensive previous experience with this DSP and associated development environments.

5.4 Choice of Development Board

5.4.1 Lyretech Signalmaster Development Board

The initial development board chosen, in January 2000, was the Lyretech Signalmaster with an ADSP21062 SHARC and Xilinx XC4000 FPGA [145] on-board. The option to add a further 4 SHARCs and a frame-grabber provides excellent flexibility. Development work was started on the basic platform, with various routines implemented to test all major components. However, this early testing highlighted three major problems with the platform: SHARC fault, communication reliability and software driver errors. The SHARC hardware fault although difficult to diagnose was simple to rectify, with a replacement chip solving all problems. The software drivers provided, both for the Signalmaster platform and the host PC, were in relatively early stages of release and had some minor errors that effected functionality. In particular, host PC software was unstable over extended use. Full development code for all Signalmaster software was provided and suitable modifications and workarounds were developed to allow use of the platform.

After the first two problems had been resolved, further testing with longer image sequences was performed. It was at this stage that a problem with the reliability of the RS232 host PC communication became obvious; previously the SHARC hardware fault may have masked this problem. As the volume of data transferred increased errors started to appear randomly through

the data. Occasionally, the errors would be in the uploads of control / information data rather than the download of image data. After consultation with Lyretech the communication was moved from the RS232 port to the Ethernet port. However, this failed to solve the problem with errors still a serious problem within a 100-frame sequence. Despite extensive work it was not possible to guarantee a reliable communication channel to the Signalmaster and the decision was taken to move onto an alternate platform.

5.4.2 Bittware Hammerhead DSP Development Board

In November 2000 the SHARC platforms available were almost unchanged. The Bittware Hammerhead quad-SHARC PCI development board provided all of the functionality required and an early delivery could be guaranteed, so this platform was selected. This allowed all of the planned development work to take place and the platform could be used in the final Dervish system if required, although overall cost may be a problem. Figure 5.1 gives a simplified diagram of the internal structure for the Hammerhead development board. The board has four SHARCs and SDRAM linked around a central bus, which is controlled by the SHARC*fin* ASIC. The SHARC*fin* ASIC controls external access, such as the host PCI bus. The SHARC processors are also connected in a ring structure through their high-speed Link Ports, allowing direct processor-to-processor communication. A more detailed description of the Hammerhead development board is given in Appendix D.

The ADSP21160 used on the Hammerhead is the most recent and powerful variant of the Analog Devices SHARC at the time of writing. Figure 5.2 provides a simplified diagram for the internal structure of the ADSP21160 SHARC processor. The three independent data buses can be seen, along with the dual-ported internal SRAM, which is divided into two 2Mbit blocks. The secondary computation unit is a mirror of the primary and is used when performing SIMD operations. A detailed description of the ADSP21160 SHARC processor, including an overview of the instruction set, is provided in Appendix D.

There are a number of different SHARC processors varying in price and performance; a recent addition to the SHARC family is the TigerSHARC, which allows MMX-style SIMD operations on 8-bit and 16-bit information and may be of interest to the final system. Due to the high compatibility of code between different types of SHARC processors the preferred platform for the final system will simply be the lowest cost solution that fulfills the performance requirement.

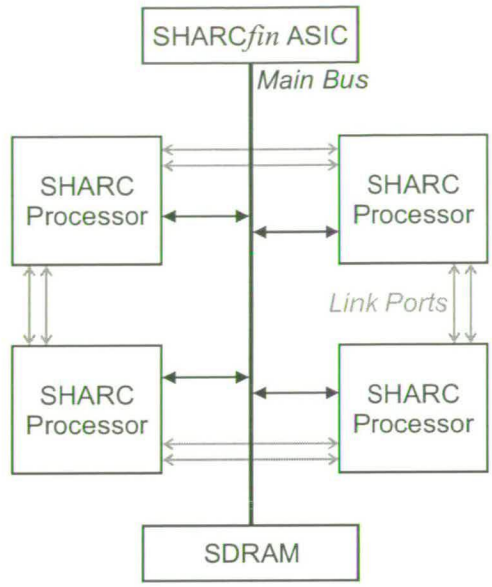


Figure 5.1: *A simplified diagram of the structure of the Bittware Hammerhead quad-SHARC development board. The SHARCs and SDRAM are all connected through a central bus, which is controlled by the SHARCfin ASIC. The SHARCfin ASIC controls external access as well, such as the host PCI bus. The SHARC processors are also connected in a ring through their high-speed Link Ports, allowing direct processor-to-processor communication.*

5.5 DSP Design Issues

5.5.1 Host-DSP Transfer Environment

The Hammerhead development board is a PCI board that can plug straight into the Dervish embedded PC. Similarly a frame-grabber could be plugged into the Dervish embedded PC and data transferred across. Alternatively, the Hammerhead has two PMC+ sites and a frame-grabber could be plugged into one of those. However, for testing and development the host PC will control and download all data. Since this setup is for testing only no attempt to optimize performance has been made.

The host PC test program required was written in Microsoft Foundation Classes through Microsoft Visual C++. At this stage a simple Host-DSP transfer protocol was used. Due to the nature of the Bittware libraries provided it is simpler to perform all data transfer operations from the host side and to maintain the signal variables on the DSP side. The major advantage to this system is that it simplifies the coding task on the DSP side, where code development is considerably more difficult. The DSP side of the data transfer system must simply wait for the signal variable to indicate that the data has been download to the external memory buffer. Effi-

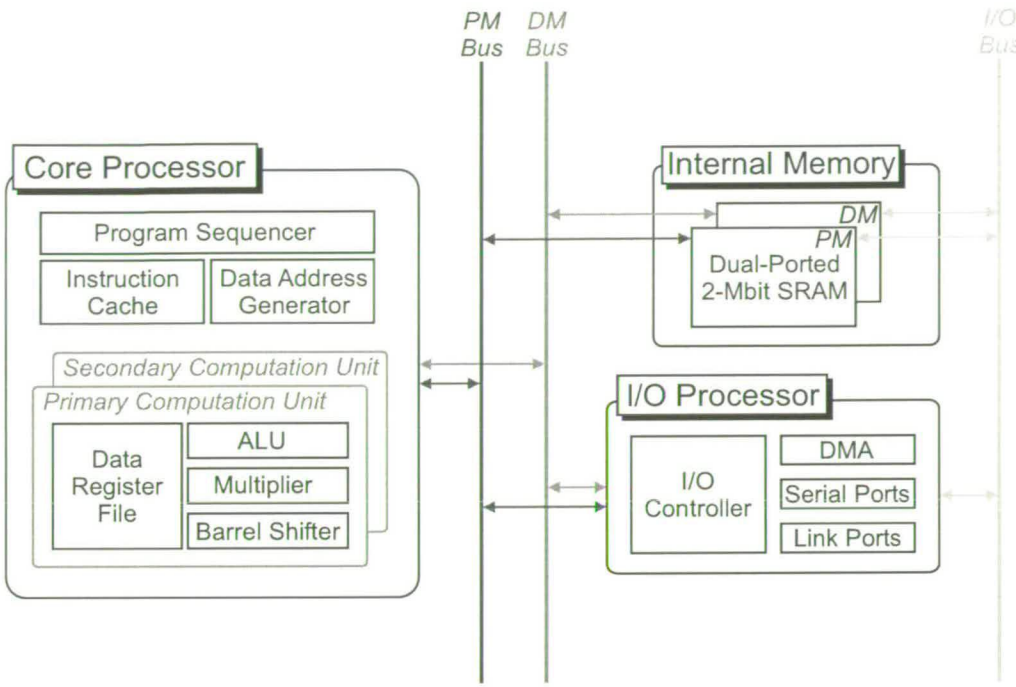


Figure 5.2: *A simplified diagram for the structure of the ADSP21160 SHARC processor. The three data buses (PM, DM, I/O) can be seen, along with the dual-ported internal SRAM, which is divided into two 2Mbit blocks. The secondary computation unit is used when performing SIMD operations.*

cient transfer from the external memory buffer to the internal memory buffer, using the DMA controller within the I/O processor can be utilised. The complete Host-DSP transfer environment is given in Appendix D, including the routines developed to allow DMA transfer between memory banks.

5.5.2 Data Transfer Format

The ADSP21160 SHARC is a 32-bit floating-point processor and the computation units can handle either 32-bit fixed-point or floating-point operations. It does not have SIMD operations for 16-bit or 8-bit operations, although a final system choice may select the TigerSHARC which can perform these operations. The use of interpolation will mean that real numbers will be required in computation. It is common practice to use integer operations, scaled up to allow a few decimal places, because of efficiency on general-purpose microprocessors. However, the SHARC has single cycle operations for all floating-point operations, including format conversion which can include IEEE unbiased rounding. For these reasons, floating-point operations will be used.

The image sequences will probably be in 8-bit data format but the SHARC does not have a native 8-bit data format. While it would be possible to pack the data, reducing transfer demands on the SHARC I/O processor, the relevant unpacking process could be a significant burden. Instead, the SHARC allows two 16-bit integers that have been packed into a 32-bit integer to be unpacked with absolutely no extra computational cost. The top 16-bits of the integer are sign-extended for signed numbers or zero-padded for unsigned numbers. Therefore, the two options are software unpacking of 8-bit data or hardware unpacking of 16-bit data. The decision here is really concerned with in-place or out-of-place calculations. If an extraction (copy) operation is already taking place (out-of-place) then the software unpacking is less of a overhead. However, if calculations are taking place on the data directly within the image (in-place) the software unpacking could be a serious problem. Section 5.5.6 discussed the issues of in-place and out-of-place calculations further and a decision on data format is made at that point.

5.5.3 DSP Clock Speed and Timer

The ADSP21160M SHARC processors on board the Bittware Hammerhead development board operate at 80MHz, although the target speed for the processor is 100MHz in the future. To help aid development and optimization it is important to gain accurate information on the performance characteristics of the algorithms being tested. The SHARC processor contains a timer, in the program sequencer, that counts the clock cycles taken between initialisation and each subsequent read. The timer counts down from a given number, and causes an interrupt when it reaches zero; the longest time it can count is approximately 240 seconds.

Therefore, the timer was used extensively throughout code development and all SHARC performance figures are quoted in clock cycles in this thesis. The Bittware Hammerhead development board contains no operating system, although real-time kernels are available, allowing very accurate timing of code segments. In practice, repeat timings of a specific code segment varied by a few clock cycles when Host-DSP data transfer was involved. Furthermore, it was determined that the routines to initialise and read the timer required 25 clock cycles. For these reasons all timings presented in this thesis have had 25 clock cycles removed and have then been rounded to the nearest 10 clock cycles.

5.5.4 Zero-overhead Loops

The SHARC allows use of zero-overhead loops, where a loop structure is defined within the program sequencer and then allowed to run with no explicit termination check or branch instruction. An example is shown, in Algorithm 1, in which a load-accumulate loop is executed 1000 times.

```
lcntr=1000,     | do(pc,_END-1)until lce; |  || start of loop
r15=dm(i1,m6);  |                         |  || data load
F1=F1+F15;      |                         |  || accumulate
_END:           |                         |  || end of loop
```

Algorithm 1: *Zero-overhead loop demonstration with a load-accumulate loop executed 1000 times.*

However, the types of loop termination are limited to status flat checks on the loop counter or program registers; threshold checks are not allowed in this style of loop. In Algorithm 1 the check is on the *loop counter equal-to-zero*, shown as `lce`. In order to allow optimum operation the loop should be kept compact, to allow all instructions to be stored in the instruction cache, and all context switches should be avoided.

5.5.5 SHARC Memory Structure

The ADSP21160 is a 4-Mbit SHARC with the memory split into two independent 2-Mbit memory blocks, called the program memory (PM) and data memory (DM). Both memory blocks have an independent data bus connected to the core-processor and are dual-ported to the I/O data bus. The SHARC code is stored in a special code segment (48-bit) and the data memory can be either 32/40/64-bits. The different data types can be mixed within each memory bank but must be stored in their own segment. Appendix D provides more information on the memory format. The program code must be stored in the PM but the size of the code segment in the PM can be reduced leaving room for extra data. This is useful for two reasons: firstly, image processing is very memory intensive and, secondly, it allows dual data accesses where two data values can be loaded, or written, in one cycle. Algorithm 2 shows a dual-data access loop where a constant value is subtracted from a sequence of elements D_n. At each cycle of the loop the previous data element D_{n-1} is written, the current data element D_n is subtracted and D_{n+1} is loaded from memory.

Dual data access is complicated by the fact that the PM bus is normally used to fetch instruc-

`lcntr=1000,`	`do(pc,_END-1)until lce;`		*start of loop*
`F1=F8-F6,`	`r8=dm(i2,m4),`	`pm(i12,m8)=r1;`	*Sub N, Ld N+1, Wr N-1*
`_END:`			*end of loop*

Algorithm 2: *Dual data access demonstration where a constant value is subtracted from a data sequence.*

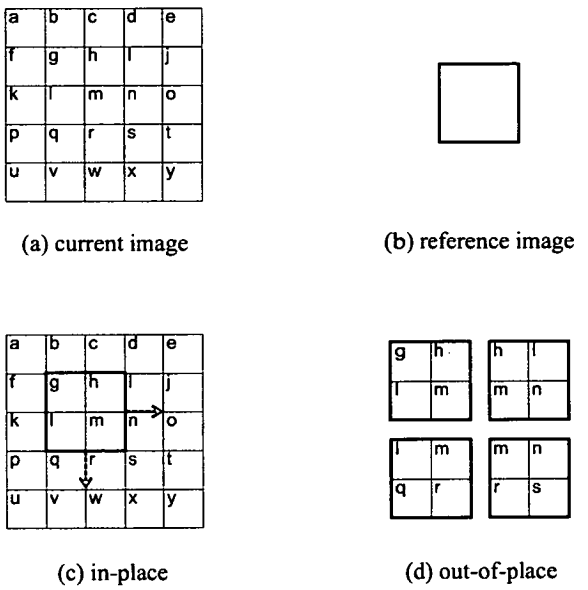
tions. If the instruction is memory then the dual data access operation will perform correctly. If the instruction is not in memory then the whole operation will take two cycles instead of one cycle. Furthermore, the data memory access must utilise address registers from the first data-address-generator, which are numbered `I1-I7` and `M1-M7`. Similarly, the program memory access must utilise address registers from the second data-address-generator, which are numbered `I8-I15` and `M8-M15`. There are no restrictions on the source / destination register `Rx`.

5.5.6 Regions: in-place or out-of-place?

One area of correlation-based feature tracking that presents an immediate and interesting problem is computation of similarity measure (or equivalent). On a general-purpose computer the cost of moving regions of memory is very-high, unless specific cache optimization has taken place. Therefore, when calculating the similarity measure between a reference region and a region from the new image it is likely that the region from the new image will be extracted throughout the calculation (in-place). In contrast, it would be possible to perform an explicit extraction operation and then the similarity measure (out-of-place).

The image data is two-dimensional and must often be treated as such. However, with calculations such as the similarity measure the image data can be treated as one-dimensional. This will allow use of a single zero-overhead loop over the entire image and avoids repeated position calculations at the end of each row (normally resetting the column position and adding one to row position). Figure 5.3 demonstrates the two types of calculations.

In the case of the SHARC DSP it is not clear if either in-place or out-of-place calculations have an advantage and so both options were tested with the SAD, which is the simplest similarity measure calculation. The results are shown in Table 5.1 and it is clear that in-place calculations are slightly quicker. However, the correlation motion model proposed in Section 6.3 will allow more efficient use of out-of-place regions. Furthermore, operating with a more complex similarity measure the out-of-place calculations will become more efficient and this is addressed



(a) current image (b) reference image

(c) in-place (d) out-of-place

Figure 5.3: *The current image (a) and reference image (b) are shown. In the in-place calculation the reference image is compared at each of the position within the image. However, in the out-of-place calculation a specific region is extracted for each position and then each region is compared with the reference region.*

further in Section 6.6. For these reasons, all coding will take place based around out-of-place regions utilising an optimized region extraction routine.

Method	SAD Performance
in–place	3450 cycles
out–of–place	4182 cycles
Ratio	0.83

Table 5.1: *Results for a performance comparison of in-place and out-of-place region calculations using the SAD similarity measure.*

The design of the optimized region extraction routine is relatively straightforward but is important to remember the structure of the zero-overhead loops. In particular, the basic nature of the terminating conditions. To meet this criteria it is necessary to perform the loop operation based on the region size, which is a fixed width. Two index variables can then be set-up, with the zero-overhead loop incrementing the region index (*extracting to*) and an explicit instruction incrementing the image index (*extracting from*). The use of out-of-place regions also allows the use of 8-bit data packing format with the optimized region extraction routine performing the unpacking. Using the hardware 16-bit unpacking, the final 8-bit unpack becomes a relatively simple task using the hardware shifter to extract the two values.

5.6 Summary

This chapter started off by reviewing the algorithm constraints, previously presented in Chapter 3, and their effect on the hardware choice. Section 5.3 examined possible hardware platforms and the Analog Devices SHARC DSP chosen as the basic processing unit. In the future it is anticipated that the Platform FPGA's being developed by Xilinx and Altera will provide a higher-performance solution at lower costs. However, at the time this decision was made the lack of good system integration, particularly I/O, would have dramatically increased development time. Section 5.4 detailed the choice of development board. The initial choice of development board was the Lyre Signalmaster but this proved unreliable and so a Bittware Hammerhead board was employed with quad-ADSP21160 DSPs. The specific design issues with the Bittware Hammerhead board and ADSP21160 SHARC are briefly discussed in Section 5.5. Appendix D provides more detailed information on all the hardware issues.

Chapter 6 goes on to identify key parameters in the correlation-based feature tracking algorithm using the Dervish field-trial data sets to measure accuracy and select the final parameters. Chapter 7 introduces feature management and a passive navigation harness to the feature tracking system and uses the simulated image sequences to measure accuracy over an extended period. Comparison with other motion estimation algorithms are presented.

Chapter 6
Correlation-based Feature Tracking

6.1 Introduction

The previous chapters have presented a review of image processing techniques for estimation of ego-motion for vehicle navigation and a review of possible hardware platforms for real-time implementation. Having considered the constraints imposed by the Dervish landmine-clearance vehicle both a motion estimation technique and hardware platform have been identified.

Correlation-based region tracking techniques satisfy the main criteria for practical robot vision [146], that of robust and reliable operation, and were selected for this reason. The iterative calculations across a small region maps into the hardware structure of the SHARC platform chosen but the high computational cost of correlation-based tracking remains an obstacle to practical use.

This chapter presents extensions to other correlation-based feature tracking techniques designed to maintain the accuracy required for the Dervish landmine-clearance vehicle while improving the performance. The key extension required is to allow the correlation-based region tracking to cope with the continual rotation of the Dervish. The performance improvements come from two areas: application specific enhancements, where prior knowledge about the application provides areas for optimization, and platform specific enhancements, where SHARC specific optimization can take place.

The first stage is an analysis of the motion for the Dervish landmine-clearance vehicle in Section 6.2, which is necessary for the optimal design of the rest of the motion estimation system. Section 6.3 takes the results from the motion analysis and applies them to the hierarchical search (Figure 3.9), describing some novel adaptions to the search structure. Section 6.4 considers the choice similarity measure, specifically given the SHARC platform and Section 6.5 analyses the the interpolation technique required by the correlation motion model.

The manually tracked Dervish field-trial data sets were used throughout this chapter, as they are real world data for the target application. Many of the system parameters are chosen throughout

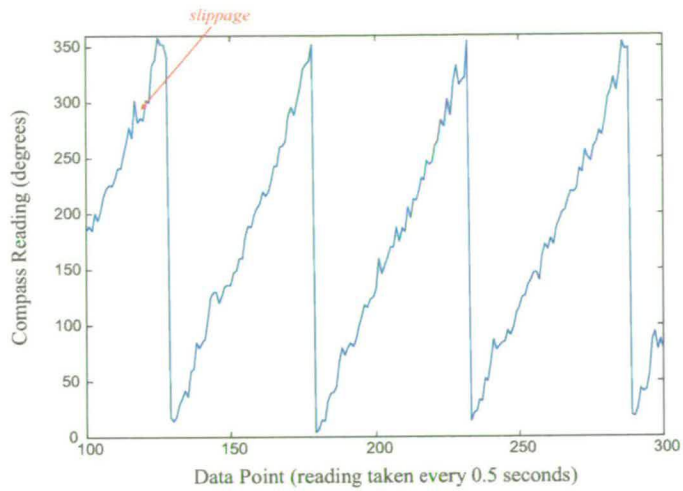


Figure 6.1: *Telemetry data taken from an electronic compass mounted on-board Dervish.*

this chapter. Unless otherwise stated the search radius for testing is ± 24 pixels square and 12 degrees, at one degree increments, shown normally as $(\pm 24, \pm 24, 12)$. The choice of search strategy and radius is discussed in Section 6.3. The extraction of rotated regions, including interpolation, is discussed in Section 6.4. A block size of 27 pixels square was chosen for testing, unless otherwise stated; the choice of this block size is discussed further in Section 6.5. It should be noted that all techniques are pixel accurate and, as such, a certain degree of quantization can be seen in the mean error and error variance for each sequence. For testing at this stage the features have been positioned to allow tracking throughout the entire sequence without moving out of the field of view. Alternative positions were tested to confirm that results present a reasonable view of overall accuracy. The issues of feature management and passive navigation are addressed in Chapter 7. All tests were carried out at 5 frames per second, the minimum defined for real-time operation, unless otherwise stated.

6.2 Dervish Motion Analysis

The Dervish has a very unusual movement pattern, as discussed in Section 2.2, and this presents the motion estimation with a unique problem, namely, the rotational speed exceeds the translational speed. For this reason, careful analysis of the Dervish motion pattern was performed using the video sequences and telemetry data gained in numerous field-trials. These results are based on the video-camera set-up used in the field-trials; the video camera set-up was described in detail in Section 2.3.

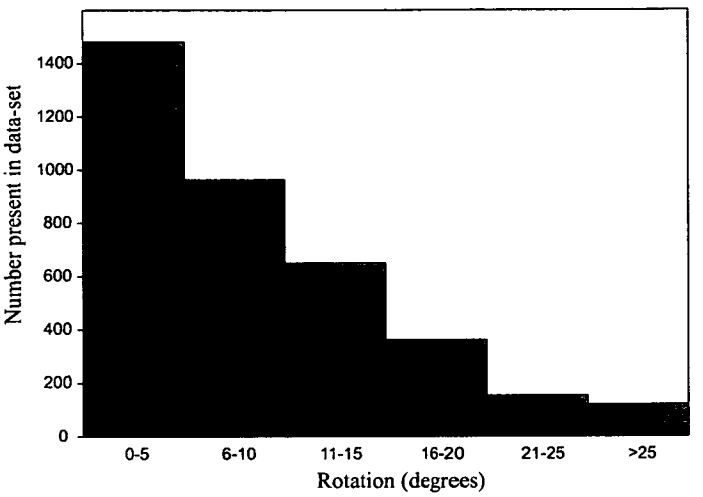


Figure 6.2: *A histogram of rotation speeds from the Dervish telemetry data. It can be seen that the compass reports a considerable proportion of large rotations (>20 degrees).*

Visual analysis of the telemetry data and video sequences from the Dervish field-trials confirmed that a tracked feature is likely to rotate between 90 and 360 degrees while in view. However, due to the wide range of working conditions and the relatively poor traction of the wheels the rotation can be fairly unpredictable. Manual estimation of the translation gives an average translation per second of 2cm and a maximum rate of 6cm per second; given the current camera set-up this translates to an average translation of 40 pixels per second and a maximum translation of 120 pixels per second.

Figure 6.1 shows telemetry data taken from an electronic compass mounted on-board Dervish. The compass takes a reading every $\frac{1}{2}$-second. It is possible to see some larger movements amongst the overall motion with one known case, where a wheel lost traction and slipped back down, marked on the graph. However, the on-board compass is not designed to be particularly accurate. This is due, in part, to the vibration of the Dervish, while in motion, and much of the compass motion is incorrect. Figure 6.2 shows a histogram of rotation between compass readings. A large proportion are greater than 20 degrees, which would suggest that the Dervish could rotate at 80 degrees per second or greater, which is not possible. Therefore, the exact nature of the rotation seen in the telemetry is not considered further and the output from the compass is not used to aid motion estimation. However, both the long-term averaged compass readings and manual tracking of the video sequences report an average rotational speed of about 25 seconds per rotation, which matches the design and is assumed to be correct.

The analysis of Dervish field-trials video sequences also confirmed the presence of ground

plane zoom and tilting. However, the Dervish is only designed to operate in terrain where every wheel can maintain contact with the ground at all times. If the wheels lose contact they may miss a landmine. For this reason, the Dervish has very stable tri-wheel design with a span of approximately 4 metres from the centre to each wheel and each wheel is approximately 1 metre in diameter. Therefore, any ground plane zoom or tilting will mainly be due to the limited suspension of the Dervish. In the video sequences the amount of ground plane tilting was extremely small and is very difficult to detect at all. For this reason, it was decided that the presence of ground plane tilting should not be accounted for in the motion estimation. The video sequences showed some ground plane zoom, up-to to a maximum change of 10%. It is not clear if this level of zoom will have a significant adverse effect on the accuracy, it must be considered in testing and accounted for if necessary.

6.3 Extending the Correlation Motion Model

6.3.1 Correlation Motion Model

In standard correlation tracking the motion model accounts for translation only but extensions have been proposed to allow both complex region transforms and complex motion. For extremely accurate estimation of optical flow across a scene, where complex multiple motions exist it may be preferable to have techniques such as the Control Grid Interpolation system proposed by Sullivan and Baker [51] or the affine warping technique proposed by Sefredis [52].

However, in downward looking ego-motion estimation the depth variation is small and projection models, such as perspective projection, are not necessary. Furthermore, the motion does not contain significant warping or multiple objects allowing all motion to be restricted to a planar rigid-body. Therefore, simpler motion models can be used. The three most commonly used models in this case are the translation only, the affine transform and the rigid-body transform (alternatively called the bilinear transform) [70].

A translational model is the simplest and subsequently the most computationally efficient for use in correlation-based tracking. The transform is defined as follows, where $\mathbf{d}$ is the displacement vector:

$$\mathbf{x}' = \mathbf{x} + \mathbf{d} \tag{6.1}$$

Rigid-body motion extends the translational model to allow rotation and scaling while conserving all angles and relative distances between points. This model restricts the type of motion that can be detected; in particular, it does not allow tilting of the ground plane. The transform is defined as:

$$\mathbf{x}' = \kappa_{scale} \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \mathbf{x} + \mathbf{d} \tag{6.2}$$

where κ_{scale} is a scaling factor, θ is the angle of rotation and $\mathbf{d}$ is the displacement vector.

Affine motion is the most general transform of the three and only preserves parallelism between lines but not the distances or angles. While the affine transform allows any possible motion of the ground plane it also allows warping that cannot easily be mapped into the true motion observed by the camera. The transformation is defined as:

$$\mathbf{x}' = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \mathbf{x} + \mathbf{d} \tag{6.3}$$

where $\{a_{11}, a_{12}, a_{21}, a_{22}\}$ define the affine warping and $\mathbf{d}$ is the displacement vector.

It has been shown that if complex motion exists in the sequence utilising a motion model capable of tracking it can significantly increase the accuracy of the motion estimation [51]. However, Shi and Tomasi [79] state that at an individual region level it is only necessary to search for translation and that the inclusion of a more complex model actually decreases the motion estimation accuracy. They argue that their standard region size, roughly between 7×7 and 15×15 pixels square, does not allow accurate detection given the very small amounts of rotation, zoom and shear seen between consecutive frames. Furthermore, the motion parameters are not independent, thereby allowing errors in the affine parameters ($\{a_{11}, a_{12}, a_{21}, a_{22}\}$)

to effect the accuracy of the displacement vector ($\mathbf{d}$).

The standard trade-off within motion model is between accuracy of the motion model versus its computational cost. However, there are two further complications on this trade-off. Firstly, the observations by Shi and Tomasi [79] demonstrate that increasing motion model complexity will only increase accuracy until a certain break-point after which the motion model will begin to hinder the motion estimation. Secondly, the unusual motion pattern of Dervish does not allow the normal assumption in ground plane navigation of mainly translational motion to be made.

Sequence	**Rotation**		**No Rotation**	
	XY (pixels)	**θ *(deg)***	***XY (pixels)***	**θ *(deg)***
Concrete	2.6 (1.9)	1.3 (0.7)	29 (540)	48 (900)
LeavesStable	4.6 (4.9)	6.3 (9.9)	LOST	
LeavesUnstable	6.6 (9.8)	3.5 (5.6)	LOST	
GrassRuler	8.5 (44)	4.6 (14)	42 (850)	62 (1400)
Grass	9.1 (39)	3.9 (11)	60 (1400)	51 (910)

Table 6.1: *The Concrete sequence is tracked with and without a rotation search space. The mean error in pixels is shown (error variance is in brackets) for both X-Y and rotation accuracy. It can be seen that the tracking without rotation produces very large errors after a short number of frames and in this case it has completely lost the target being tracked.*

In fact Dervish rotates continually as part of its motion pattern and was designed to complete a rotation about every 25 seconds, which was confirmed in field-trials. Assuming the motion estimation is operating at 5 fps, a realistic minimum for correct operation defined in Chapter 2, the rotation between frames will be approximately 3 degrees. Therefore, any region being tracked will become redundant after more than ten frames because the rotation will mean that the majority of pixels will be matching in the wrong place. Table 6.1 shows all the Dervish field-trial sequences tracked with and without rotation in the motion model, showing a clear advantage to tracking with rotation in the motion model.

There are two possible approaches to solving this problem: rapid replacing of the reference region or integration of rotation in the motion model. As part of an investigation into reference region updating [124, 147] different update rates were tested. It is clear from the results, shown in Table 3.1, that updating too often causes the accuracy of the track to degrade. It can also be seen that never updating the reference region can cause the track accuracy to degrade. In most of the test sequences $T_s = \infty$ proved effective but the optimal update rate varied in testing

between $T_s = 1$ and $T_s = 25$ depending on the sequence.

Therefore, while rapid updating of the reference region appears to be a simple solution it will increase the speed at which estimation errors are embedded, making the system more like a dead-reckoning sensor and reducing the overall accuracy. By using the same reference region over a longer period and artificially rotating it any error from the current estimate is not embedded into the overall reading until the reference region goes out of view, which should only be after considerable rotation.

At this stage, due to the very small amount of zoom and shear seen in the image sequences the motion model will be restricted to rotation and translation (rigid-body without zoom). It is possible that the introduction of zoom and / or shear may be necessary but this is to be balanced with the large increase in computation required.

6.3.2 Extended Rotation Correlation Algorithm

Normally, the motion model is embedded directly into the search structure for the correlation tracking, treating the extra parameters as extra dimensions to be searched. There are three costs associated with the extended correlation search: calculation of the similarity measure, extraction of transformed regions (which includes interpolation) and calculation of the hierarchical pyramid. Normally a single reference region is compared at every location, which would require both the extraction and correlation to be performed at every location. The total cost for this naïve implementation of the extended rotation correlation algorithm (ERCA) is given by:

$$\textit{Naïve ERCA Cost} = (rn^2) \cdot (EC) \tag{6.4}$$

where E is the cost of extracting a transformed region and C is the cost of calculation for the similarity measure. However, after some consideration of the application it is possible to dramatically reduce this.

The cost of extracting a transformed region is considerable, especially when the interpolation is included. However, because the search is restricted to rotation and translation (rigid-body without zoom) it is possible to move the rotation over from the current image to the reference image. In this case a set of rotated reference regions are compared at every translation point in

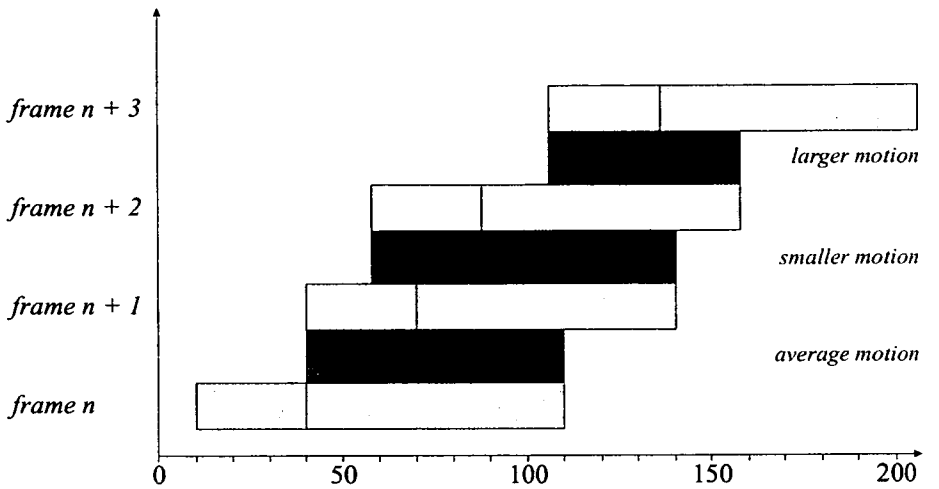


Figure 6.3: *The blue sections show the areas of overlap between consecutive reference rotation sets. The overlapping areas can be reused helping to reduce the computational cost.*

the current image. This is analogous to the obvious optimization of placing the rotation search as the outer-most loop, allowing every extraction of a rotated region to be used in the translation search before moving on to the next rotation step.

However, the main advantage in having the reference region rotated is that much of the original rotation set of extracted reference regions may be reused in the next frame, saving a significant number of extraction. As previously stated the Dervish rotates at approximately 15 degrees per second but motion in the field-trials has been up to 40 degrees per second for very short periods. Therefore, a rotation search of 60 degrees per second (-20 to 40 degrees from current angle) will capture the unpredictable motion seen in field-trials. Under the rotational average of 15 degrees, the vast majority (approximately 75%, depending on angle quantization) of the rotation set will be present in the next rotation set. Therefore, the average cost will be approximately $\frac{1}{4}rE$, dramatically improving over the naïve implementation of rn^2E.

This extended correlation search must now be embedded into the standard hierarchical search, shown in Figure 3.9. It is at this stage that we can reduce the number of times the similarity measure has to be calculated. Again the naïve cost for this is given by Equation 6.4. The use of a hierarchical search will reduce the number of times the similarity measure has to be calculated to $\sum_m r_m n_m^2$ where r_m and n_m are the rotation and translation searches, respectively, at each level (m) of the pyramid. Using the method given in [87] for identifying sub-sampling rates with a block size of 27 pixels and a search radius of 120 pixels per second, a two-level pyramid with subsampling rate of 3 is selected. This was also tested empirically and found to give the

highest level of accuracy. In particular, a higher degree of subsampling tended to remove the small amount of structure present in difficult terrain, such as grass or tarmac. Therefore, the number of times a similarity measure would have to be calculated would be $r_2 n_2^2 + r_1 n_1^2$.

Finally it is necessary to determine how to perform the distribution of the rotation search within the hierarchical search because the resampling of the image does not have an effect on the rotation search area. The obvious solution is to perform a coarse search through the full rotation search space, increasing the angle quantization. In this case the coarser angle quantization must be determined.

Table 6.2 shows the effect of rotation on the reference region given different block sizes under a subsampling rate of 3. It can be seen that a rotation of roughly three times is required to achieve the same effect, suggesting that a coarse search through rotation may be appropriate with the unit angle quantization multiplied by the subsampling rate. However, given the large amount of noise present in the input sequence, it is reasonable to assume that any average change of less than 0.1 pixel will be entirely lost within the noise; in fact, even an average change of 0.1 pixels is very small.

Region Size	Nearest Neighbour	Bilinear
$L_1 = 27$	2.00°	0.56°
$L_2 = 9$	6.00°	1.67°
$L_1 = 21$	2.50°	0.71°
$L_2 = 7$	8.00°	2.16°
$L_1 = 15$	3.50°	1.00°
$L_2 = 5$	10.00°	3.06°

Table 6.2: *The rotation in degrees required to create an average motion of 0.1 pixels in a reference region of a given size. L_2 values are the subsampled reference regions in the hierarchical structure.*

On the basis of these results, it is not clear if it is possible to accurately detect small levels of rotation in the L_2 level of the pyramid. In fact, it has been shown [79, 87] that the more complex motion model can reduce the accuracy of the individual motion estimates. Table 6.3 shows the results for the normal hierarchical search and one without rotation in L_2 and it can be seen that using a translation only search in the L_2 level actually increases the accuracy. However, because no rotation search will be performed at the L_2 level, the entire rotation search will have to be performed at the L_1 level.

Sequence	Hierarchical Search		no L_2 rotation	
	XY (pixels)	θ *(deg)*	*XY (pixels)*	θ *(deg)*
Concrete	2.6 (1.9)	1.3 (0.7)	2.6 (2.1)	1.3 (0.6)
LeavesStable	4.6 (4.9)	6.3 (9.9)	4.5 (4.8)	5.8 (11)
LeavesUnstable	6.6 (9.8)	3.5 (5.6)	6.5 (7.1)	3.4 (5.4)
GrassRuler	8.5 (44)	4.6 (14)	8.5 (65)	4.4 (29)
Grass	9.1 (39)	3.9 (11)	8.0 (36)	3.4 (8.4)

Table 6.3: *The Dervish field-trial image sequences are tracked using a normal hierarchical search algorithm and a variant which does not detect rotation in L_2 of the pyramid. The variant with no rotation tracked in L_2 presents slightly higher accuracy.*

6.3.3 Three-Level Search for Translation and Rotation

The use of a hierarchical search normally makes the use of complex search patterns redundant because of the small search spaces at each level of the pyramid. However, an extension can be applied to the 1-D search first proposed by Chen *et al.* [148] to avoid the increase in computational complexity at the L_1 level, discussed in Section 6.3.2. In the original 1-D search the requirement is that overall surface structure is monotonic in both dimensions (UESA). This requirement is very strict and often violated, causing the original 1-D search to be inaccurate. However, instead of applying a vertical/horizontal alternating search a translation/rotation alternating search is proposed.

The proposed algorithm is shown in Figure 6.4 and can be best analysed by breaking the algorithm down and looking at the requirements for each of the three stages. The first step is a translational search at level L_2 which limits the translation to a 3×3 area. As previously discussed, this stage does not lose accuracy due the lack of rotation. The second stage performs a rotation minimisation with a subsampling applied to the angle quantization but not to the image resolution. Essentially, the L_1 rotation reference set is used but only every 3^{rd} reference region is tested. The third stage starts with a search space identical to the standard hierarchical search, and performs a minimization for rotation and translation at L_1.

Therefore, the main requirement for this algorithm is that the rotation can be accurately identified when the translation is only known to ± 2 pixel accuracy. Since no sub-pixel translation is detected the smallest motion achievable is 1-pixel. However, referring back to calculations used in Table 6.2, it can be seen that to achieve an average pixel motion of 1-pixel it is necessary to rotate a considerable number of degrees; for example, a 15-pixel block with bilinear

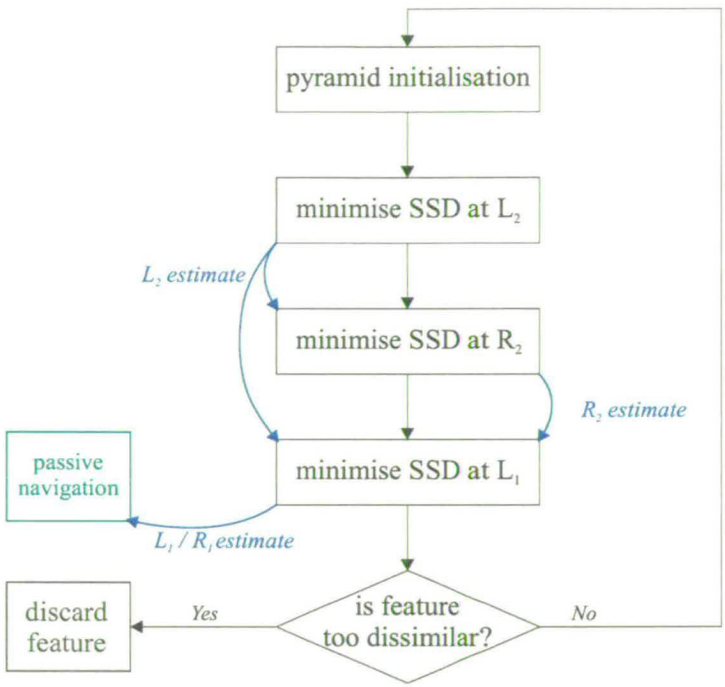


Figure 6.4: *A simplified flow-chart for the proposed correlation-based tracker. The blue curved lines show how data is used from each of the estimates, where L_n estimates are translation and R_n estimates are rotation.*

interpolation requires a 10 degree rotation and even a 27-pixel block requires 5 degree rotation. Therefore, the effective quantization is much lower for rotation than translation. This can be seen in Figure 6.5, which shows the rotation search at the position of best match for a single frame in the Concrete sequence, with a well-behaved monotonic error surface that has relatively small changes between points.

Table 6.4 shows the angle of rotation identified as the best match given an error in the translation position. It can be seen that within $\pm$ 1 pixel of the true translation the rotation can be correctly identified within $\pm$ 3 degrees. This table used frame 20 from the Concrete sequence with a 27-pixel block started with its centre at (225,144). Therefore, the rotation search behaves sufficiently well to be approximated as unimodal when the translation is close to the position of best match allowing the proposed extended rotation correlation algorithm to perform correctly. The three-level search has a lower cost than the normal hierarchical search strategy with the number of times a similarity measure would have to be calculated reducing from $r_2 n_2^2 + r_1 n_1^2$ to $n_2^2 + r_2 + r_1 n_1^2$.

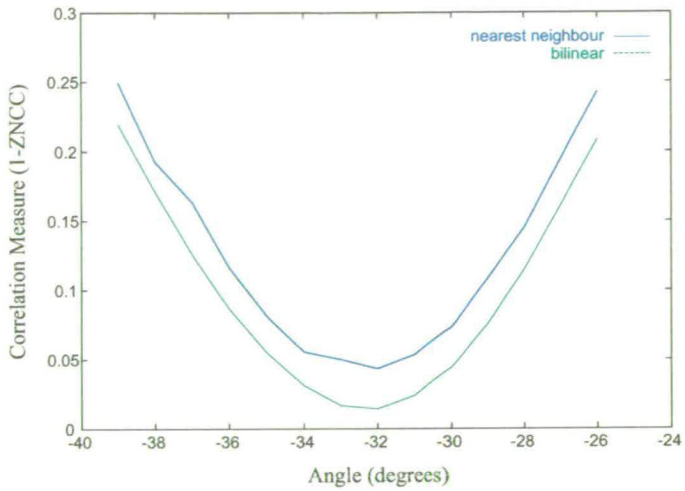


Figure 6.5: *The rotation search at the correct translation for a single frame of the Concrete sequence showing a smoother curve with bilinear interpolation. However, both methods of interpolation produce a well-behaved monotonic error surface in a local area around the true translation.*

6.3.4 Resampling Kernel

One more hierarchy parameter that remains to be chosen is the resampling kernel used to calculate the hierarchical pyramid. There are a number of different techniques to calculate the hierarchical pyramid but the most popular is the Gaussian pyramid used by Kanade-Lucas-Tomasi[87] and Anandan [61]. This has been employed in all test results up to this stage. This kernel can reasonably be extended to any width but the computation is relatively expensive due to the convolution requiring a multiplication per pixel. Figure 6.6 gives an example of a 5×5 Gaussian resampling kernel.

0.0014	0.0090	0.0168	0.0090	0.0014
0.0090	0.0576	0.1067	0.0576	0.0090
0.0168	0.1067	0.1979	0.1067	0.0168
0.0090	0.0576	0.1067	0.0576	0.0090
0.0014	0.0090	0.0168	0.0090	0.0014

Figure 6.6: *A normalized Gaussian resampling filter with kernel width of 5.*

An alternative is the mean filter, or box filter, which is the simplest convolution kernel. An

		X Error				
		−2	**−1**	**0**	**1**	**2**
Y Error	**−2**	26	26	26	27	32
	−1	33	32	32	32	31
	0	35	35	35	34	34
	1	38	36	36	38	27
	2	38	39	39	39	26

Table 6.4: *The table shows the angle of best match under varying amounts of translation error (in pixels). The true angle is 35 degrees, correctly identified when there is no translation error. It can be seen that within ±1 pixel of the true translation the rotation can be correctly identified within ±3 degrees. This table used frame 20 from the Concrete sequence with a 27-pixel block.*

example is shown in Figure 6.7. The filter is computationally very efficient and there are implementation methods for the mean-filter that can improve performance over a naïve implementation [146]. However, the filter has a significant low-pass filtering effect and can blur edges or corners excessively.

$1/9$	$1/9$	$1/9$
$1/9$	$1/9$	$1/9$
$1/9$	$1/9$	$1/9$

Figure 6.7: *A mean-filter (box-filter) with kernel width of 3.*

Resampling with a Gaussian resampling kernel is probably the most common method for hierarchical image processing and gives excellent results. However, both filters are tested due to the significance of the performance benefit. The resampling kernel is also used to smooth the full-resolution (L_1) image before any feature selection or tracking occurs.

6.3.5 Accuracy

At this stage there are two viable options for the rotation search, using either a normal hierarchical search or the three-level search proposed in Section 6.3.3. The rotation reference set proposed in Section 6.3.2 can be used in both these search strategies. There are also two different resampling kernels that need to be tested for accuracy.

However, first it is necessary to use the minimum operating frequency of 5 frames per second, defined in Section 2.2.3, to select basic search parameters for the hierarchical search. As previously stated the maximum speed for translation is 120 pixels per second and at 5fps this gives ± 24 pixels per frame. Therefore, the L_2 translation search must be ± 8 pixels and, in theory, the L_1 search range should be ± 1 pixel. However, in practice a significant overlap provides the system with a possible way to recover any inaccuracy within the L_2 match. For this reason the L_1 translation search is empirically determined.

The rotation search range is defined as -20 to 40 degrees per second and at 5fps this gives a rotation search space of 12 degrees (-4 to 8 degrees). The optimum unit angle quantization is addressed in Section 6.6 and for the time being is assumed to be 1 degree. In practice, with a radian based mathematics library a search range of -0.06 to 0.14 radians with a quantization angle of 0.02 radians is employed. For both the standard hierarchical search and the three-level search the first rotation search is set as -0.06 to 0.12 radians with a quantization angle of 0.06 radians. The second rotation search is then defined as -0.04 to 0.04 radians with a quantization angle of 0.02 radians.

Table 6.5 shows the results from tracking with a normal hierarchical search compared to the three-level search. The basic search parameters defined above were used with a 27-pixel block. The L_1 translation search radius was set at ± 1 pixel. It can be seen that the tracking is relatively accurate apart from the two grass sequences, which are very difficult to track due to lack of good texture. The L_2 matching was not accurate enough and with no recovery mechanism the final position match was occasionally incorrect. This would probably cause the feature to be discarded and a new one selected, which reduces the active lifespan of a feature. Within the grass sequences the lifespan of the feature is reduced to approximately 4 seconds.

A possible solution for the grass sequences would be to increase the L_1 translation search area to provide a degree of overlap. Table 6.5 also contains results for an overlapped hierarchical search pattern where the L_1 search radius was set to ± 4 pixels. It can be seen that a significant accuracy improvement is made. However, the number of times the similarity measure will have to be calculated almost trebles. This trade-off between accuracy and efficiency presents a fundamental choice within the entire system: track more features with lower accuracy or track less features with higher accuracy. This problem is addressed fully in Section 6.6 when the complete trade-off between accuracy and efficiency is known.

Sequence	Hierarchical Search		Three–Level Search		Overlapped Hierarchical	
	XY (pixels)	θ *(deg)*	*XY (pixels)*	θ *(deg)*	*XY (pixels)*	θ *(deg)*
Concrete	4.8 (24)	3.3 (23)	3 (4.5)	2.7 (11)	2.6 (1.9)	1.3 (0.7)
LeavesStable	6.6 (15)	7.1 (20)	7.6 (24)	6.6 (17)	4.6 (4.9)	6.3 (9.9)
LeavesUnstable	6.3 (13)	4.3 (8.2)	6.5 (7.1)	3.4 (5.4)	6.6 (9.8)	3.5 (5.6)
GrassRuler	14 (130)	13 (120)	15 (140)	7.8 (41)	8.5 (44)	4.6 (14)
Grass	16 (10)	8.2 (35)	15 (210)	15 (141)	9.1 (39)	3.9 (11)

Table 6.5: *The Dervish field-trial sequences are tracked using different search patterns and parameters. The overlapping hierarchical search provides the highest accuracy but is computationally expensive.*

The two resampling kernels, Gaussian and mean, were also tested for accuracy. Table 6.6 shows the accuracy of tracking for the two kernels when used for image smoothing and image resampling. The kernel width was 5 for the Gaussian kernel and 3 for the mean kernel. The sub-sampling rate was 3 for both kernels. The overlapped hierarchical search has been employed. It can be seen that neither resampling kernel provides significantly higher accuracy. Therefore, the mean kernel will be preferred due to its lower computational cost.

Sequence	Gaussian Kernel		Mean Kernel	
	XY (pixels)	θ *(deg)*	*XY (pixels)*	θ *(deg)*
Concrete	2.6 (1.9)	1.3 (0.7)	2.5 (1.6)	1.1 (0.4)
LeavesStable	4.6 (4.9)	6.3 (9.9)	4.7 (4.9)	6.2 (14)
LeavesUnstable	6.6 (9.8)	3.5 (5.6)	7.5 (12)	4.1 (6.3)
GrassRuler	8.5 (44)	4.6 (14)	6.7 (34)	3.5 (11)
Grass	9.1 (39)	3.9 (11)	9.9 (57)	5.0 (17)

Table 6.6: *The two resampling kernels, Gaussian and mean, are tested on the Dervish field-trial data sets for tracking accuracy. The overlapped hierarchical search strategy is employed. No significant difference can be seen between the two resampling kernels, suggesting that the mean-filter should be preferred because of its lower computational cost.*

The results from Section 6.3 demonstrate that the hierarchical tracking with a mean resampling kernel can provide significant performance improvement while providing accuracy in the tracking. At this stage the overlapping hierarchical search strategy can be identified as the most accurate and the three-level search can be identified as the most computationally efficient.

6.4 Similarity Measure

6.4.1 Choice of Similarity Measure

There is a wide variety of similarity measures, as presented in Chapter 3. However, apart from highly-specific situations no similarity measures have been shown to significantly exceed the performance of the standard measures, such as the SAD, SSD or correlation coefficient. Furthermore, there is no strong evidence in the literature that any of the standard measures performs consistently better than the others over an extensive data set. Table 6.7 shows the accuracy of the three main similarity measures tested on the general purpose data sets from Chapter 4, providing a total of almost 500 frames. It can be clearly seen that none of the similarity measures consistently outperforms the others.

Sequence	SAD	SSD	ZNCC
CarParkB	6.2 (8.8)	6.0 (8.3)	4.2 (6.3)
Cyclist	2.0 (1.4)	2.1 (1.9)	1.8 (1.3)
Pedfollow	2.8 (25)	1.9 (5.0)	1.5 (0.6)
Snowseq	20 (180)	18 (100)	24 (200)

Table 6.7: *The table demonstrates that on general data sets the three main similarity measures do not have a consistent or significant difference in accuracy. The mean error is in pixel with the error variance in brackets.*

The SAD has proven very popular for block matching due to its low computational cost and amenability to hardware implementation. The SSD provides a similar low-cost alternative but it requires a multiplication per pixel making it considerably less suitable for hardware implementation. However, the SAD and SSD both vary with illumination changes and it has been reported that this can reduce their accuracy [65]. Any illumination change is likely to be global in downward looking ego-motion estimation, especially considering the small size of regions being tracked. Therefore, it is likely that the correlation minimum will still be correctly identified. However, a disadvantage to illumination-variant similarity measures is that a change in illumination can effect the confidence, or dissimilarity measure, which is very important to long-term motion estimation.

To reduce the effect of illumination changes use of a correlation coefficient has been proposed. Pearson's R [30] is the most common correlation coefficient and is sometimes referred to as the Zero-mean Normalized Correlation Coefficient (ZNCC) [149]; it is given below in two different

forms:

$$ZNCC = \frac{\sum_x \sum_y \left(\mathcal{I} - \overline{\mathcal{I}}\right)\left(\mathcal{R} - \overline{\mathcal{R}}\right)}{\sqrt{\sum_x \sum_y \left(\mathcal{I} - \overline{\mathcal{I}}\right)^2 \times \sum_x \sum_y \left(\mathcal{R} - \overline{\mathcal{R}}\right)^2}} \tag{6.5}$$

$$ZNCC = \frac{\text{cvar}(\mathcal{I} - \overline{\mathcal{I}}, \mathcal{R} - \overline{\mathcal{R}})}{\sqrt{\text{var}(\mathcal{I} - \overline{\mathcal{I}}) \times \text{var}(\mathcal{R} - \overline{\mathcal{R}})}} \tag{6.6}$$

Although the calculation is represented by variance and and co-variance it is not necessary to normalize each calculation as the two divisions cancel each other out. It is, on first examination, significantly more complex than the SAD and SSD and requires greater numerical accuracy when performing the computation. However, high-performance DSPs have single-cycle floating-point, or fixed-point, operations available on multiple computation units, rendering the traditional computational cost invalid. Therefore, it is necessary to analyse each of the three similarity measures for accuracy and computational performance for this particular application.

Two other variations that should be addressed are the box-filter SAD and KLT SSD method. The box-filter SAD notes that, within translation, consecutive blocks share many of the same calculations and a partial-sum calculation pattern can be applied to reduce the overall number of absolute-difference calculations. However, this does not map into the SHARC architecture for two reasons: there is an increase in the memory required and, more importantly, the SHARC performs most efficiently when the SIMD computation units is used to perform a repetitive calculation without writing back. This is discussed in more detail in Section 6.5.3. The KLT method, presented in Chapter 3, performs a Newton-Raphson minimization on the error surface, reducing the average number of calculations that must be performed. Again, however, this does not allow the SIMD computation units to be utilised to best effect. Furthermore, the minimisation can never be as accurate in all cases as simply performing the SSD. For these reasons, the KLT could be considered if a completed system performs with sufficient accuracy but requires a reduction in the cost of the similarity measure.

6.4.2 Accuracy

The three different similarity measures are used almost interchangeably when applying correlation-based tracking, with different authors reporting slight advantages to each. Therefore, they were

tested on the Dervish field-trial data set using the overlapped hierarchical motion model developed in Section 6.3 and the results are shown in Table 6.8.

Sequence	SAD		SSD		ZNCC	
	XY (pixels)	θ *(deg)*	*XY (pixels)*	θ *(deg)*	*XY (pixels)*	θ *(deg)*
Concrete	2.1 (1.0)	1.5 (1.6)	2.4 (1.2)	1.2 (0.6)	2.5 (1.6)	1.1 (0.4)
LeavesStable	4.7 (4.0)	6.2 (15)	4.7 (4.6)	6.1 (14)	4.7 (4.9)	6.2 (14)
LeavesUnstable	6.8 (9.2)	3.8 (5.0)	7.5 (12)	4.2 (6.8)	7.5 (12)	4.1 (6.3)
GrassRuler	7.2 (34)	3.8 (11)	7.3 (34)	3.9 (11)	6.7 (34)	3.5 (11)
Grass	9.2 (66)	3.7 (8.6)	8.9 (55)	3.5 (6.5)	9.9 (57)	5.0 (17)

Table 6.8: *Investigation into the accuracy of the three main similarity measures on the Dervish field-trial data sets. The mean error for translation is in pixels and the mean error for rotation is in degrees; the error variance is in brackets. It can be seen that there is no consistent difference in the accuracy between the three similarity measures.*

There is no consistent difference in the accuracy of each of the three different similarity measures on the field-trial data sets. The translation and rotation accuracy are shown and in general high accuracy in one results in high accuracy in the other. Therefore, the choice of similarity measure for the Dervish landmine-clearance vehicle depends entirely upon the computational performance on the target platform. For this a detailed analysis of the similarity measure and search algorithm is required. When reviewing the algorithm it is important to consider the inclusion of rotation set, which reuses the same rotation reference block at different translations.

6.4.3 Performance Optimization

The calculation of the Sum of Absolute Difference requires that two data values are loaded, subtracted, the absolute taken and accumulated. The related Mean Absolute Difference function, which is often quoted, requires a divide by a constant number (pixels in the block), making the divide redundant. The SHARC DSP has the absolute function as a native operation and it takes one clock cycle but the subtraction, absolute and accumulate operations must be performed in separate clock cycles as they all use the ALU. The data load can be pipelined, loading the data in the first pass and calculating with that data in the second pass, with the ALU operations, as shown in Algorithm 3.

This means that the calculation time is approximately 3-times the number of data points. For a 27-pixel block the measured calculation time is 2220 clock cycles (140 clock cycles for pro-

```
r8=dm(i4,m6);                                    preload data from i1/i4
r12=dm(i1,m6);                                   m6 is read-increment
lcntr=729,        do(pc,_END-1)until lce;        start of loop
F2=F8-F12,        r8=dm(i4,m6);                  subtract & data load
F15=abs F2,       r12=dm(i1,m6);                 ABS & data load
F1=F1+F15;                                       accumulate
_END:                                            end of loop
```

Algorithm 3: *Sum Absolute Difference SHARC implementation.*

cedure construction and destruction). The calculation of the Sum of Squared Differences is almost identical to the calculation of the SAD, however, the absolute operation is replaced by a multiplication. In this case, it is not possible to use the multiply-and-accumulate to perform the square operation because the SHARC requires the multiply operands to come from specific sets of registers in different halves of the register bank. Therefore, the calculation time is, also, 2200 clock cycles.

The SAD and SSD are both in the correct format for implementation using the SHARC SIMD architecture, which involves setting the internal mode correctly and from then on every data-based operation (e.g. read, write, ALU) is performed on two consecutive 32-bit values. When directed to use SIMD on the loop the compiler is capable of generating efficient code and the calculation time for both SAD and SSD using the SIMD mode is 1150 clock cycles. This is a performance improvement of 1.92, which is very close to the theoretical value of 2.

The calculation of the ZNCC is rather more complicated and it is necessary to break down the calculation for most efficient implementation. Starting with Equation 6.6 it can be seen that the calculation for two individual blocks can be broken down as follows:

1. Calculate the mean for the block
2. Subtract the mean from every pixel in the block
3. Calculate variance for the block
4. Calculate co-variance between blocks and divide by variance of each block

The mean calculation involves a load / accumulate loop that can be efficiently implemented as a one-cycle loop on the SHARC. The subtraction loop requires a load, subtraction and write-back which is implemented by the compiler with 2 cycles per pixel. However, it is possible to

read from one memory block and write-back to the other while performing an ALU operation; essentially Write N, Sub N+1, Load N+2 in a single instruction. To make use of this the compiler partially unrolls the loop to interleave two calculations per loop giving 1.5 clock cycles per loop. However, it is possible to move the first load and subtraction out of the loop and then start the loop from the second data element, reducing the loop to 1 clock cycle per pixel. The code is shown in Algorithm 4

```
m4=2;                                                          Set read increment
m8=m4;                                                         And write increment
r8=dm(i2,m4);                                                  preload data N
F1=F8-F6;;                                                     pre-subtract data N
r8=dm(i2,m4);                                                  preload data N+1
lcntr=365,      do(pc,_END-1)until lce;                        start of loop
F1=F8-F6,       r8=dm(i2,m4),           pm(i12,m8)=r1;         Sub N+1, Ld N+2, Wr N
_END:                                                          end of loop
```

Algorithm 4: *SHARC optimized routine for the subtraction of the mean (F6) from block but the data is read from DM (i2) and written to PM (i12) to allow an extra level of optimization.*

The variance calculation can be rotated round, like the subtraction calculation, to preload the first data value outside the loop. This allows the data load to be in parallel with the multiplication. However, the multiplication / accumulation cannot be performed in parallel because the SHARC requires that the parallel multiply / ALU operations use specific sets of data registers and the passing to/from these registers takes extra clock cycles. Therefore, a single variance calculation takes 2 clock cycles per pixel. However, it is possible to perform the variance calculation for 2 blocks simultaneously and in this case the register passing instructions can be interleaved within the code, reducing the computation time to 3 clock cycles per 2 pixels. The SHARC code is shown in Algorithm 5.

The co-variance calculation is efficiently implemented by the compiler and results in a two cycle loop, where the first data load is in parallel with the multiplication and the second data load in parallel with the addition.

Using the functional breakdown it is possible to see that the first three steps need only be calculated once per block and it is only the fourth step that needs to be performed at every match position. Any correlation search that includes more than just translation will require repeated use of a single block. This allows the first three steps to be pre-calculated for every block and the fourth step can be repeated at every location where the block is used.

```
m0=2;
r2=dm(i4,m0);
r4=pass r2,     r0=dm(i2,m0);
F12=F2*F4,      F9=F9+F12,          m1=m0;
r7=pass r0;
lcntr=365,      do(pc,_END-1)until lce;
F12=F0*F7,      F9=F9+F12,          r2=dm(i4,m0);
r4=pass r2,     r0=dm(i2,m1);
F12=F2*F4,      F11=F11+F12,        r7=r0;
_END:
```

Set read increment
Preload A
Preload B
Calc A (MAC)
Register pass for B
start of loop
MAC B, Ld A
Pass A, Ld B
MAC A, Pass B
end of loop

Algorithm 5: *SHARC optimized routine for calculating the variance of two blocks simultaneously. The interleaving of the calculation masks the register passing time and allows greater parallelism of instructions.*

Step	Name	Computational Cost		Frequency
		Non–optimized	Optimized	
1	Mean	810 cycles	410 cycles	*per block*
2	Subtraction	1115 cycles	590 cycles	*per block*
3	Variance	1115 cycles	590 cycles	*per block*
4	Co–variance	1490 cycles	790 cycles	*per location*
1,2,3,4	Total (measured)	9020 cycles	4040 cycles	

Table 6.9: *Computation time for optimized SHARC routines used in the calculation of the correlation coefficient. Non-optimized calculation times are shown for comparison. The first three steps can be performed once for each block while the fourth step must be performed at every comparison location.*

It can be seen from Table 6.9 that the computation time for the first three steps is 1590 clock cycles per block and the computation time for the fourth step is 790 clock cycles. Therefore, the computational cost in clock cycles for the different similarity measures is given by:

$$SAD/SSD = 1150 \cdot (n^2 r) \tag{6.7}$$

$$ZNCC_{naive} = 4040 \cdot (n^2 r) \tag{6.8}$$

$$ZNCC_{opt} = 1590 \cdot (n^2 + r) + 790 \cdot (n^2 r) \tag{6.9}$$

where n is the translation search space and r is the rotation search space. $ZNCC_{opt}$ refers to the optimized version where the first three steps are pre-calculated. Assuming the FSA

is employed for a search space of $(\pm 7, \pm 7, 15)$ and optimized SIMD SHARC code is used the computational cost for both the SAD and SSD is 3881250 clock cycles. In contrast, the computational cost for a naïve implementation of ZNCC that performs the entire calculation at every step would be 13625000 clock cycles, more than three times greater. However, for the optimized calculation of the ZNCC the computational cost reduces to 3047850 clock cycles, which is significantly lower than the SAD or SSD. It should be noted that the computational performance scales linearly with the size of the reference block down to a minimum size of 6×6.

6.4.4 Application to Different Search Algorithms

Although the performance optimizations allow the ZNCC to perform well under the FSA different search algorithms are common and few perform as many calculations using a relatively small number of blocks. In the example of a $(\pm 7, \pm 7, 15)$ search space, there are 240 blocks used in 3375 calculations. Using a hierarchical search the translation search area at each level of the pyramid is dramatically reduced, making this modification less efficient. Using Equations 6.7 and 6.9 it can be seen that the stage at which the ZNCC becomes more efficient is the zero-crossing, given by:

$$\begin{aligned} 0 &= ZNCC_{opt} - SAD/SSD \\ 0 &= \left[1590 \cdot \left(n^2 + r\right) + 790 \cdot \left(n^2 r\right)\right] - \left[1150 \cdot \left(n^2 r\right)\right] \\ \left(n^2 + r\right) &= 0.22 \left(n^2 r\right) \end{aligned} \tag{6.10}$$

Figure 6.8 shows the relative efficiency curves under different translation search spaces; the rotation search (r) is set at 8(red), 15(green) and 25(blue). The stage at which the ZNCC becomes more efficient (i.e. the zero-crossing) varies with the rotation search radius but a 2 pixel translation search is more efficient with the ZNCC given the likely search ranges for this application.

The effectiveness of the ZNCC optimization will depend entirely on the search space and traversal strategy used but it has been shown that the overall computational cost for the similarity measure must be looked at specifically for the target hardware platform and on the SHARC

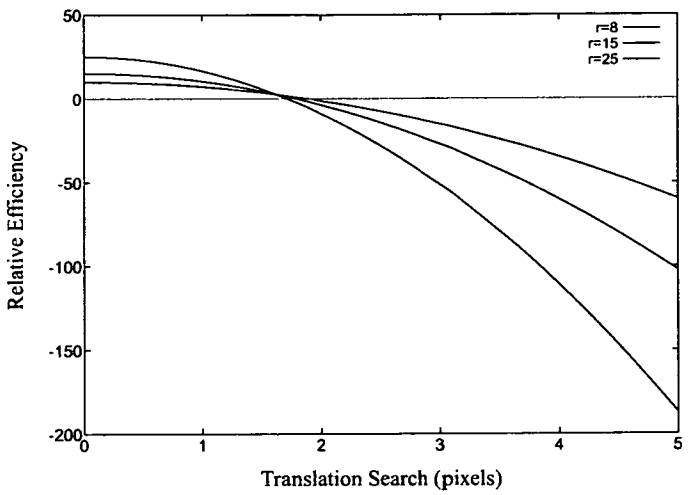


Figure 6.8: *The relative efficiency of the ZNCC and SAD similarity measures are shown for varying translation search sizes (n); a positive value indicates the SAD is more efficient, while a negative value indicates the ZNCC is more efficient. The rotation search (r) size is set at 8 (red), 15 (green) and 25(blue).*

platform the ZNCC can be an efficient choice. For the overlapped hierarchical search pattern, discussed in Section 6.3, a two-level hierarchical pyramid is used with a L_2 translation search space of ± 8 pixels and a L_1 translation search space ± 4 pixels. The rotation search space is discussed further in the next section but at this stage the L_2 rotation search is -3 to 6 degrees with a rotation step of 3 degrees and the L_1 rotation is -2 to 2 degrees with a rotation step of 1 degree. Therefore, the computational cost in clock cycles, without use of a rotation reference set, for the SAD and ZNCC are:

$$\begin{aligned} SAD &= 650400 \\ ZNCC_{opt} &= 635500 \end{aligned}$$

The use of a reference rotation set will not effect the cost for the SAD because no pre-calculation is performed. However, for the ZNCC the number of rotated reference cores that have to be pre-calculated can be reduced to $\frac{1}{4}$ of the normal amount. Therefore, the average computational cost for the ZNCC with the use of a rotation reference set is:

$$ZNCC_{opt} = 630100$$

The rotation reference set is primarily designed to make the extraction of rotated regions more efficient and the impact at this stage is merely useful but not dramatic. It is, therefore, equally efficient to the use the SAD or ZNCC under the overlapped hierarchical search pattern. The computational cost for the ZNCC is significantly lower than would have been expected and this is largely due to the architecture of the SHARC processor. The combined effect of the similarity measure and search pattern is discussed in Section 6.6.

6.5 Extraction of Rotated Regions

6.5.1 Calculation of the Rotation Transform

The transform calculation is based upon the rigid-body rotation calculation (Equation 6.2), although no scaling is required with the extended rotation correlation algorithm ($\kappa_{scale} = 1$). This simplifies the transform down to a translation and rotation. However, the extended rotation correlation algorithm splits the rotation and translation, so only rotated regions will have to be extracted. Optimization of this transform is described below. A major variable in this is the choice of interpolation technique and Section 6.3 used nearest neighbour and bilinear interpolation in testing. Bilinear interpolation is commonly applied to correlation-based tracking but few results have been published into the effect of interpolation of the accuracy of correlation tracking.

There is a wide variety of interpolation techniques in the literature starting with zero-order (nearest neighbour) interpolation [150] through to high-order polynomial surface fitting algorithms designed for specific tasks [151, 152]. There are a number of general-purpose higher-order interpolation techniques, such as bicubic spline interpolation or bicubic convolution. Cubic splines have proven useful in interpolation in many different situations [153–155] and are normally presented as the next step up from bilinear interpolation, providing a more accurate representation of the underlying data and reducing the low-pass filtering (blurring) seen with bilinear interpolation. However, the incoming data is of poor quality, from both the camera noise and vibration, and has already undergone a low-pass filtering operation during the im-

age pre-processing. Therefore, it is unlikely that higher-order techniques will be able to give a significantly more accurate representation of the data and the increase in computational cost is very large, despite attempts to improve the computational performance [156].

For these reasons, the real-time operation requirement combined with relatively poor image quality considerably reduces the number of relevant techniques. The techniques that are discussed below are: nearest neighbour and bilinear. The inclusion of interpolation is to allow detection of rotation and is not for sub-pixel accuracy.

Nearest Neighbour (First-order) Interpolation

This simple interpolation technique rounds the true co-ordinate to the nearest sample boundary; in our case the real value is rounded to the nearest integer boundary. It is important that the rounding mode is unbiased. If the mode was slightly zero-biased, for example by setting the 0.5 value to zero, it would create a slight shift in the region position towards zero that could affect the accuracy of the results.

Bilinear (Second-order) Interpolation

Bilinear interpolation performs a linear weighting between the four closest samples to the true co-ordinate. The weight for each sample is based on the distance from the true co-ordinate. Much smoother images are generated with this technique but the total computational cost is higher and the output is essentially low-pass filtered. The basic computation for an interpolated pixel P_i at co-ordinates (x_r, y_r), given the four surrounding pixels $(P_{11}, P_{12}, P_{21}, P_{22})$ is:

$$\begin{aligned} X_n &= \text{floor}(x_r) \\ Y_n &= \text{floor}(y_r) \\ P_i &= (1 - Y_n)\left[X_n P_{12} + (1 - X_n) P_{11}\right] + (Y_n)\left[X_n P_{22} + (1 - X_n) P_{21}\right] \end{aligned} \quad (6.11)$$

6.5.2 Accuracy

The accuracy of an interpolation technique is normally judged by the visual accuracy of the reconstructed image. However, in this case the most important criteria is the accuracy of the

motion estimation achieved. The accuracy can be divided into two categories: individual frame (sub-pixel) and extended usage. In ego-motion estimation the important category is extended usage. The three remaining factors affecting the accuracy are the block size, rotation quantization step and interpolation technique. The search pattern and similarity measure having already been discussed.

Table 6.10 shows the accuracy, in both rotation and translation, over the Concrete and Grass sequences under varying rotation quantization step sizes. The block size has been increased to 33 pixels square for this experiment because the nearest neighbour interpolation technique does not perform with sufficient accuracy at a smaller block size. The results for the simpler concrete sequence show that the rotation accuracy increases as the rotation quantization step size reduces. However, limited accuracy is gained and the translation accuracy is not significantly affected by the rotation quantization size. Within the harder to track grass sequence there is a slightly more pronounced improvement in translation accuracy as the rotation step size decreases. At this stage a minimum rotation step size of 1 degree is set but this could be increased to 3 degrees for improved computational efficiency at a only slight degradation in accuracy.

Interpolation Technique	**Rotation Step (L_1)**	**Concrete**		**Grass**	
		XY (pixels)	**θ *(deg)***	***XY (pixels)***	**θ *(deg)***
Nearest Neighbour (33 pixels)	3°	2.4 (1.9)	1.7 (1.6)	LOST	
	1°	2.3 (1.9)	1.4 (1.0)	7.6 (43)	4.5 (15)
	0.1°	2.3 (1.8)	1.3 (0.6)	7.5 (10)	4.1 (10)
Bilinear (33 pixels)	3°	2.0 (1.8)	2.2 (2.3)	8.7 (52)	5.3 (15)
	1°	2.4 (1.8)	1.3 (0.6)	8.6 (52)	5.2 (15)
	0.1°	2.3 (1.8)	1.2 (0.6)	8.1 (49)	4.6 (14)

Table 6.10: *The accuracy of the Concrete and Grass data sets using the overlapped hierarchical search and the ZNCC, under varying rotation quantization step sizes. The translation mean error is in pixels with the error variance in brackets. The rotation mean error is in degrees with the error variance in brackets. The total rotation in both sequences is over 100 degrees.*

The accuracy that the correlation tracking can achieve under different types of interpolation is as important as the computational cost of the interpolation. From the results shown in Table 6.10, it can be seen that bilinear interpolation allows greater accuracy in the estimation for both translation and rotation. This is especially apparent under coarser rotation quantization sizes. However, the associated increase in accuracy is set against an increase in computational cost

and the trade-off cannot be analysed until the computational performance is known.

6.5.3 Performance Optimization

Interpolation

For nearest neighbour interpolation, the SHARC supports IEEE unbiased rounding within its floating-point to integer conversion and only requires one clock cycle whatever rounding mode is set. The conversion to integer is always required because the co-ordinate must be used as an index to the image array in memory. Therefore, the nearest neighbour interpolation is computationally free. The rounding mode takes time to set on the SHARC but it would be possible to design part, or even all, of the program to use a rounding mode rather than ANSI C truncation, which would reduce switching between modes. The computational cost for extraction of a rotated region of 27 pixels square is 11700 clock cycles. Algorithm 6 shows the hand-optimized implementation on the SHARC.

```
r8=FIX F8                    | IEEE round
r12=FIX F12                  | IEEE round
r0=180;                      | Perform array indexing
r12=r12*r0(SSI);             |
r8=r8+r12;                   |
r8=r8+r4;                    | Add array base address
r0=dm(i1,m1);                | Read pixel
F0=float r0;                 | Convert to float
```

Algorithm 6: *Hand-optimized SHARC assembler to perform nearest neighbour interpolation demonstrating the simplicity of the internal IEEE unbiased rounding mode. Although no parallel instructions can be utilised all redundant or unnecessary operations have been removed providing a 20 percent performance improvement over the compiler.*

The bilinear calculation in the format, shown in 6.11, requires random memory access and cannot, therefore, utilise the SIMD extensions on the SHARC. In this format the extraction of a rotated region of 27 pixels square tales 50960 clock cycles, which is a substantial cost when compared to the nearest neighbour. The algorithm for the bilinear interpolation is efficiently generated by the compiler and is too long to show.

The bilinear interpolation provides considerably better accuracy and, in particular, smoothness of the correlation surface. This is particularly important for good operation of the hierarchical search patterns. Therefore, the choice between the two can be viewed as a design trade-off that

is intertwined with the remainder of the correlation parameters.

Rotation Calculation

The rotation transform, using Equation 6.2, requires a number of multiplications and additions per pixel even with optimization it still requires at least 6 additions and 4 multiplications per pixel. However, the transform is a simplified affine transform and does not alter the angles between parallel lines. This allows the transformed region to be treated as a series of parallel lines (pixel rows) stacked on top of each other. Therefore, the problem can be tackled as a line drawing problem using the optimized line drawing techniques.

The incremental line drawing algorithm pre-calculates an offset (x,y) between two consecutive points in the row, reducing the internal calculations to 2 additions per pixel for traversal of the first axis. A further calculation would be required at the start of every row to traverse the second axis. This could also be performed with an incremental algorithm, requiring a further 2 additions. The first pixel must be transformed using the complete transform to be used as a starting point for the incremental algorithms. If the transform where to be used recursively rounding error in the offsets would accumulate leading to significant inaccuracy. Similarly, if the transformed region was especially large the rounding error may accumulate. However, neither of these conditions are true and the incremental line algorithm can be applied to the transform without loss of accuracy.

To avoid floating-point calculations integer-only line drawing algorithms, such as Bresenham's [157], were developed. These move in integer increments using an error-accumulator to account for the rounding error. This type of integer only algorithm may be useful in performing nearest neighbour interpolation as no rounding from floating-point to integer would be necessary. However, because the SHARC DSP combines single-cycle floating-point operation with an in-built IEEE rounding mode there is no need to utilise an integer-only algorithm.

The search space identified in previous frames is -4 degrees to 8 degrees at 1 degree increments. In practice this suggests a search of -0.08 to 0.14 radians at an increment of 0.02 radians. This, therefore, requires 12 reference regions to be stored. However, maximum rotation between frames would require two thirds of the reference set to be recalculated and average motion would require one third of this reference set to be recalculated. Table 6.11 shows the various combinations and their respective costs.

Interpolation	Block Size	Average Cost	Maximum Cost
Optimized NN	33	47000 cycles	94000 cycles
Nearest Neighbour	33	70000 cycles	140000 cycles
Optimized Bilinear	33	210200 cycles	420400 cycles
Bilinear	33	304700 cycles	609500 cycles

Table 6.11: *The cost of interpolation for two different block combinations with both the cost under average motion and the cost under maximum motion. The optimized variants, using an incremental line drawing algorithm, are also shown.*

6.6 Accuracy-Efficiency Review

The correlation-based feature tracking has been examined in detail over this chapter and key routines have been optimized for the target hardware platform. There are many parameters to select within general correlation-based feature tracking. To start with a hierarchical search pattern using a two-level pyramid was chosen due to the massive reduction in the computational cost. After analysis of Dervish's motion the search parameters were defined for the hierarchical search and it was discovered that due to low texture in many of the image sequences an overlap at the bottom level (L_1) of the pyramid provided an improvement in tracking accuracy. An alternative three-level search was proposed that maintained the accuracy of the standard hierarchical search but reduced the computational cost. This essentially presents an option of a search with higher accuracy (overlapped hierarchical) or a search with a lower computational cost (three-level search). A 3×3 mean resampling kernel was shown to be of comparable accuracy to the common Gaussian resampling kernel for this application and the mean resampling kernel has a considerably lower computational cost.

The correlation similarity measure was examined next and it was shown that there was no significant difference in accuracy between the SAD, SSD and ZNCC. Therefore, the SAD and ZNCC were optimized on the target hardware platform. The SAD is simple to implement, and optimize, and produces the best performance in most circumstances. However, the ZNCC can be partially precalculated and the heavy reuse of rotated reference cores means that under certain search parameters the ZNCC is more efficient. Using the three-level search the SAD is more efficient and using the overlapped hierarchical search the ZNCC is more efficient.

The extraction of rotated regions was investigated next and two different interpolation routines tested across different rotation step sizes. It became clear that no further accuracy is gained

in reducing the rotation step size below 1 degree. The choice of nearest neighbour or bilinear interpolation was not clear. A rotation calculation based on an incremental line drawing algorithm was presented that provides a significant performance improvement regardless of the interpolation technique. Therefore, the correlation-based feature tracker parameters that remain to be chosen are:

- Search Pattern: three-level search or overlapped hierarchical
- Similarity Measure: SAD or ZNCC
- Block Size: 21×21, 27×27 or 33×33 pixels
- Interpolation: nearest neighbour or bilinear

The cost of the image pyramid is constant within the system and does not effect the individual features. Reference block update strategies are discussed in Section 7.2.2 but this does not effect the parameter selection at this stage. However, the rest of the associated costs all combine to effect the final system selection.

Search Pattern and Similarity Measure

The accuracy-efficiency trade-off could be approached from a number of different angles. In this case the most fundamental decision is on the search pattern. The similarity measure is fundamentally linked into the search pattern. A block size of 27 pixels with bilinear interpolation is assumed. Using an average accuracy across the five Dervish field-trial data-sets the relative efficiency of the different search patterns against the similarity measures is shown in Figure 6.9.

It can be seen from the results that the overlapped hierarchical search provides the highest accuracy but this is set against a large increase in computational cost. The three-level search provides similar accuracy to the standard hierarchical search but at a considerably lower computational cost. At this stage it is important to guarantee the accuracy of the motion estimation and the overlapped hierarchical search is clearly the most accurate. Furthermore, the extra robustness provided by the overlapping search is clearly visible in the consistency of the results presented in Table 6.5. The choice of similarity measure is very close but the ZNCC is chosen for its slightly better computational performance.

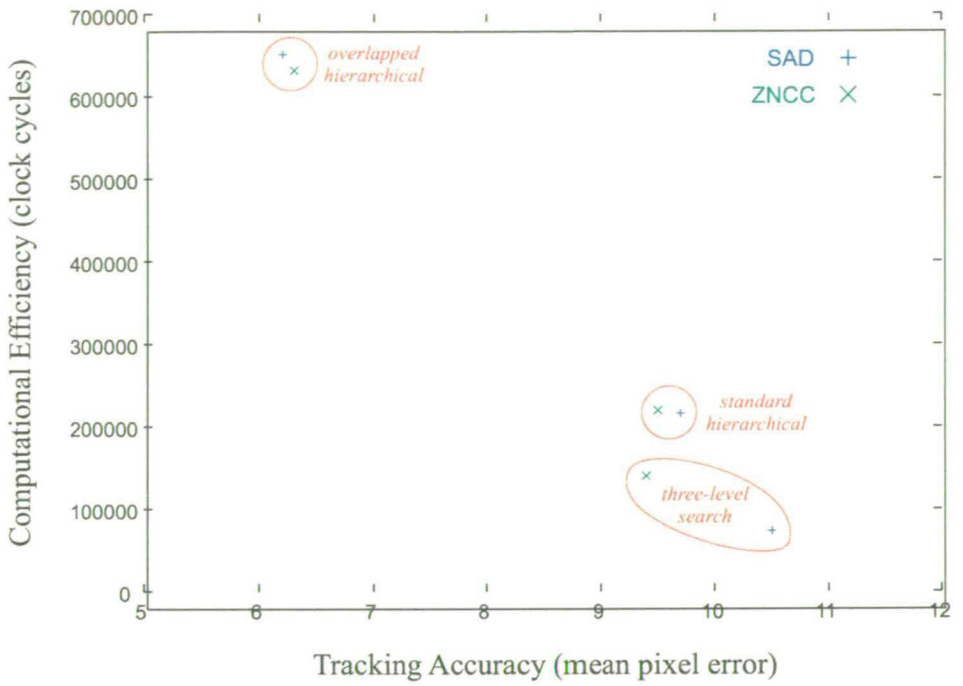


Figure 6.9: *The relative accuracy-efficiency of the three different search patterns using the two similarity measures is shown. It can be seen that the overlapped hierarchical search pattern provides the highest accuracy while the three-level search provides the lowest computational cost. The computational cost is for the search pattern only and not for the extraction of rotated regions.*

Interpolation and Block Size

Having selected the overlapped hierarchical search pattern and the ZNCC similarity measure, the only two remaining parameters are the block size and interpolation technique. Again, the average accuracy across the five Dervish field-trial data-sets is set against the relative efficiency of the different block sizes and interpolation techniques. The results are shown in Figure 6.10. The computational cost shown is for the both full algorithm, including the extraction of rotated regions and the calculation of the similarity measure. The optimized version of the rotation calculation, based on the incremental line algorithm, is used.

It can be seen from the results that the rotation accuracy increases as the block size increases. The effect of the block size on the translation accuracy is dependent on the interpolation technique, with a measurable effect under nearest neighbour interpolation but little effect under bilinear interpolation. Using nearest neighbour interpolation it would be necessary to use a 33×33 pixel block size to achieve a reasonable degree of accuracy. Using bilinear interpolation a block size of 21×21 or 27×27 pixels proves adequate. A block size of 27×27 pixels with bi-

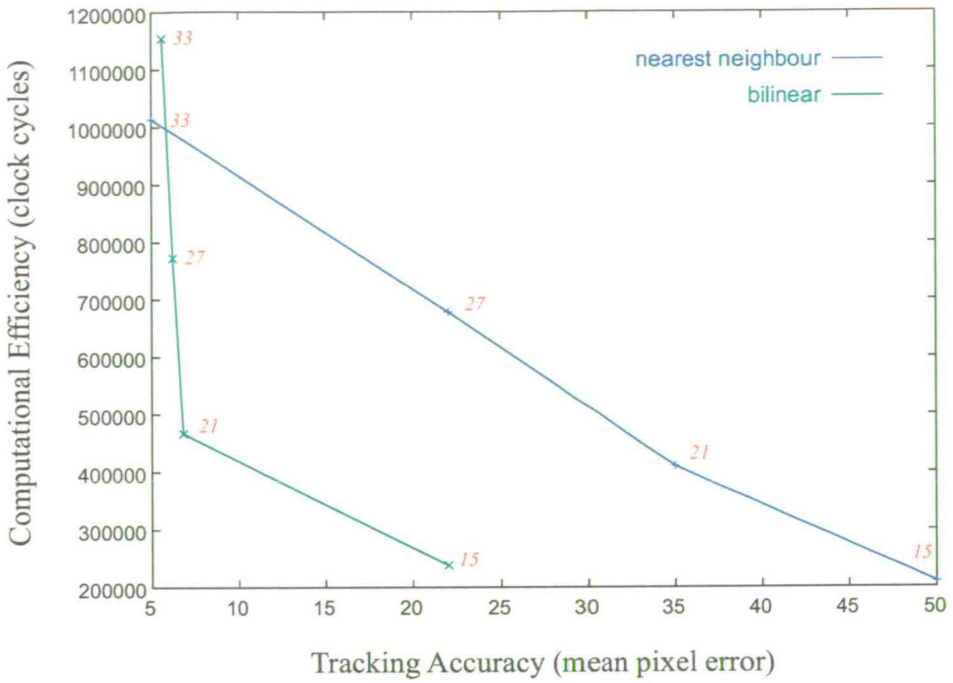


Figure 6.10: *The relative accuracy-efficiency of different block sizes and interpolation techniques is shown. It can be seen that a block size of 21×21 pixels with bilinear interpolation is required for good operating accuracy. A block size of 33×33 pixels with nearest neighbour interpolation is required for good operating accuracy.*

linear interpolation is selected because it proved more robust through testing, allowing features to be tracked for considerably longer than a block size of 21×21.

Accuracy-Efficiency Conclusions

Under the parameters for tracking an individual feature the total computational cost is 860700 clock cycles on the Hammerhead SHARC. The Hammerhead SHARC has an operating frequency of 80MHz so at 5 frames per second there is 16 million clock cycles per SHARC. It is, therefore, obvious that multiple features can be tracked in real-time on the current hardware platform with the correlation-based feature tracking parameters chosen. The average tracking accuracy for a single feature across the five Dervish field-trial data sets was 6.2 pixels error. Although a single feature is not accurate enough for the Dervish navigation system it provides a good starting point for the passive navigation, discussed in detail in the next chapter.

At this stage it is reasonable to ask questions of how much more accurate can the correlation-based feature tracking be on an individual feature. Table 6.12 shows the five Dervish field-trial

data sets tracked at 25 frames per second using the Full Search Algorithm with a block size of 33 pixels. It can be seen that practically no accuracy improvement is seen with the sequences. This is primarily because the majority of tracking errors seen within the final system design are from outside sources. One common example is vibration from the Dervish motors causing the poor focus on the digital camera leading to increased image noise. Another is when the Dervish judders, translating rapidly but returning to the original position. It is possible that these problems could present a serious problem for the final vision-based navigation system. However, most of the present problems are due to on-going mechanical development of the Dervish and field-trials after the key stages of mechanical development are complete would be necessary to determine if the Dervish artefacts remain a problem.

Sequence	**Final Parameters**		**FSA (25fps)**	
	XY (pixels)	**θ *(deg)***	***XY (pixels)***	**θ *(deg)***
Concrete	2.6 (1.9)	1.3 (0.7)	2.3 (1.8)	1.1 (0.6)
LeavesStable	4.6 (4.9)	6.3 (9.9)	4.0 (4.3)	5.4 (9.7)
LeavesUnstable	6.6 (9.8)	3.5 (5.6)	6.9 (9.9)	3.8 (5.8)
GrassRuler	8.5 (44)	4.6 (14)	7.0 (31)	3.5 (9.9)
Grass	9.1 (39)	3.9 (11)	8.5 (37)	4.0 (12)

Table 6.12: *A full-search algorithm operating at 25 frames per second is tested against the overlapped hierarchical search pattern operating at 5 frames per second. It can be seen that there is limited improvements made given the very large increase in computational cost.*

6.7 Summary

This chapter has examined the Dervish motion and designed possible hierarchical search patterns, including a novel three-level search pattern. These were tested on the Dervish field-trial data sets using the manually tracked ground truth and their accuracy and performance detailed. The two major computational costs within the search patterns were identified as calculation of the correlation similarity measure and the extraction of a rotated region. These two major costs were then examined separately. Three correlation similarity measures were tested and careful optimization of the SHARC assembler implementations took place. Using the overlapped hierarchical search from Section 6.3, the ZNCC was slightly more efficient than the SAD but this is highly dependent on the search pattern and radius. The extraction of rotated regions was examined next in Section 6.5 with the interpolation providing the bulk of the computational

cost.

Section 6.6 performed a review of the accuracy and efficiency of each of the remaining parameters for correlation-based feature tracking. After careful review, the overlapped hierarchical search pattern using the ZNCC was chosen with a block size of 27 pixels and bilinear interpolation. It was confirmed that these parameters will allow the current hardware platform to track multiple features at 5 frames per second. Chapter 7 takes the results presented here and uses them to design the vision-based navigation system on the Bittware Hammerhead platform for the Dervish landmine-clearance vehicle.

Chapter 7
Vision-based Navigation System

7.1 Introduction

This chapter finalises the details of the vision-based short-range navigation system for the Dervish landmine-clearance vehicle. Up to now the feature starting position has been selected by hand. Furthermore, all starting positions remained within the image for the duration of the test and were selected so that the target track was not lost. Section 7.2 addresses the issues of feature management by discussing and selecting suitable feature selection techniques. Chapter 4 investigated reference block update strategies and presented a novel method with low computational and memory requirements. This reference block update strategy is introduced into the target tracking application and the tracking accuracy examined. The final task for feature management is to discard features that have lost the target so that they do not have a negative impact on the passive navigation. A novel dissimilarity measure is introduced due to the inclusion of reference block updating and the accuracy examined. All tests are carried out at 5 frames per second, the minimum defined for real-time operation, unless otherwise stated.

The final task remaining for the proposed vision-based navigation system is to select a suitable passive navigation system. Section 7.3 provides a brief summary of possible passive navigation techniques before selecting a suitable technique. Section 7.4 then provides test results on the accuracy of the proposed vision-based navigation system. Some comparative results with other motion estimation strategies are also presented. Section 7.4 concludes by finalising the design system and parameters for the vision-based navigation system.

Having a completed design for the vision-based navigation system, Section 7.5 looks at the implementation details on the Bittware Hammerhead development platform. In particular, the multi-processor design issues are examined and a load distribution suggested. Section 7.5 also re-examines the chosen hardware platform and discusses which member of the Analog Devices SHARC DSP family would provide the most suitable choice for the final application. One point to note is that chosen Bittware development platform and the Analog Devices SIMD SHARC (ADSP-21160N) are both called the Hammerhead. To differentiate I have referred to

the hardware platform as the Bittware Hammerhead and the Analog Devices DSP as the SIMD SHARC throughout this chapter.

7.2 Feature Management

7.2.1 Feature Identification

The issue of feature identification, or selection, has been addressed numerous times but one of the most important to date is the Shi and Tomasi method [79] where the concept of a *good feature* is defined mathematically. This method has been commonly and successfully used in various applications. However, it is based on a mathematically good feature within their system of equations. An alternative is the Moravec interest operator [104, 158], which essentially performs the correlation search around a small neighbourhood and determines the strength of the correlation peak. All points with a local correlation variance below a certain threshold are discarded.

However, after examination of the video sequences from the Dervish field-trials and experimentation with both the Shi and Tomasi feature selection technique and the Moravec interest operator it became clear that very few points matched the criteria of being good to track. With the thresholds reduced it was possible to identify features but these had a fairly even distribution across the image. Figure 7.1 shows the Shi and Tomasi technique applied with a lowered threshold. Combining the small image size (360×288) with the large block size and the even feature distribution, it became difficult to see any advantage to the computationally costly process of feature selection.

Therefore, it was decided to test the accuracy of tracking on the five Dervish field-trial data sets at different positions. Table 7.1 shows results from the Grass and Concrete data sets with the middle point in each group representing a good feature and the remaining four points as random features. The number of frames has been reduced to cover only 5 seconds because after more than 5 seconds replacement of some features would be necessary. It can be seen that the accuracy of each feature over the short term is always high, even given a poor texture area. In fact, the majority of the image is taken up by relatively non-descript terrain and, as suggested above, it is not always possible to identify any useful features.

The Dervish landmine-clearance vehicle requires a high-degree of accuracy for the motion

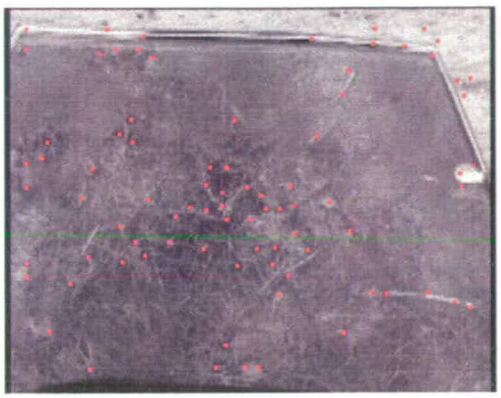


Figure 7.1: *The Shi and Tomasi feature selection technique is shown applied to the first frame from the Grass sequence. The threshold has been lowered to identify 80 features from the frame.*

estimation and a large number of tracked features provides a level of redundancy that is useful in improving the accuracy. Given that there is limited difference in the short term tracking accuracy at any position in the image it was decided to employ a dense grid structure for the reference features.

Normally a minimum distance between features would be defined to avoid overlapping features but otherwise the feature distribution would not be controlled. In the case of the Dervish it is known *a priori* that considerable rotation is contained within the normal motion and that features towards the corners of the image may move outside of view very rapidly. To avoid this scenario a minimum distance from edge and corners of image is enforced. After experimentation a border size of 24 pixels, increasing to 32 pixels for the corners, was chosen as this almost always allowed the feature to track for multiple frames. The minimum feature separation distance must be large enough to accommodate rotation and was selected at 42 pixels after experimentation. Using an even distribution of features with the parameters selected above over $\frac{1}{3}$ of the image area is tracked and with the current search radiuses, every part of the image is searched by at least one feature.

Figure 7.2 shows the proposed feature selection procedure demonstrated on the Concrete data set. It can be seen that good coverage is achieved across the entire image. On average over the five Dervish field trial data sets the number of features per frame was 42. Given that a spread of features will be employed the useful lifespan of each feature will vary widely and it is important to firstly allow good features to track for as long as possible and secondly to detect and remove redundant features. The replacing of lost features then becomes a relatively trivial

Sequence	Start Position	*XY (pixels)*	θ *(deg)*
Concrete (5 seconds)	(178, 144)	2.2 (0.5)	4.9 (2.1)
	(272, 144)	1.6 (0.7)	1.7 (0.2)
	(225, 144)	2.4 (0.5)	1.6 (0.1)
	(225, 97)	2.3 (0.4)	2.4 (0.7)
	(225, 191)	2.5 (0.4)	1.6 (0.6)
Grass (5 seconds)	(153, 215)	4.1 (6.1)	1.3 (1.8)
	(247, 215)	3.3 (1.6)	1.5 (0.7)
	(200, 215)	2.6 (2.9)	1.4 (1.3)
	(200, 168)	2.5 (2.0)	2.0 (2.2)
	(200, 115)	1.4 (0.5)	1.4 (0.9)

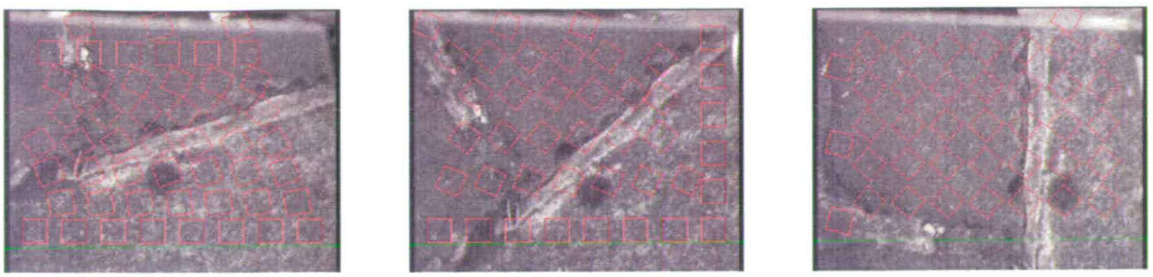
Table 7.1: *A number of different start positions tested in the Concrete and Grass field-trial data sets. The start positions cover a range of possible features from areas including areas of very high and very low texture. It can be seen that the large block size allows good tracking over the short term (5 seconds) for these two image sequences.*

task of placing as many features as possible into the grid structure while observing the position rules described above. It should be noted that the feature dissimilarity is not fully implemented in the test and some features are too close to the edge of the image. The dissimilarity measure, including feature removal, is discussed in Section 7.2.3 and these issues are corrected.

7.2.2 Feature Reference Updating

One area that has not received much attention within correlation-based feature tracking is how to update a feature. Chapter 4 presented an empirical investigation into reference block updating methods and concluded with the novel track-initiated DLS filter. It was shown that reference block updating can help improve the accuracy of correlation-based feature tracking but all strategies have an associated increase in computation. The computational performance was optimized for the track-initiated DLS filter but is still significant.

Using the track-initiated DLS reference block update strategy proposed in Section 4.4, 3 parameters are needed to design the filter: image noise variance (σ_x), sampling frequency ($1/T$) and maximum signal acceleration ($\ddot{x}$). The minimum sampling frequency for correct operation is 5 frames per second, so $T = 0.2$. The technique proposed by Rank *et al.* [127] was again used to estimate the image noise variance of the sequence and the average value, in grey-levels,



frame 26 *frame 121* *frame 166*

Figure 7.2: *The proposed feature selection procedure is demonstrated on the Concrete data sets. Three frames have been selected through the image sequence and it can be seen that good coverage is achieved across the entire image.*

was 5. The final parameter is the maximum signal acceleration and this is harder to estimate for this real world application. However, it is desirable to operate an over-damped filter in this application because accuracy will, in general, be better served through discarding of a feature that changed too rapidly. In contrast, an under-damped filter may allow a feature to be incorrectly tracked and the incorrect data incorporated into the reference block, creating a long-term data outlier within the motion estimates. Therefore, a maximum acceleration of 1 grey-level per second per second was initially selected and this was confirmed as a suitable value through empirical testing. Taking these values and using Equation 7.1 (originally given in Section 4.4) we can select θ. The solution, in this case, is $\theta = 0.90$

$$\frac{(1-\theta)^3}{(1+\theta)} = \frac{\ddot{x}T^2}{3\sigma_x^2} \tag{7.1}$$

The EMP filter used to initiate the track for the DLS filter should be operated for the first n frames. To determine the change over point Equation 7.2 should be balanced. In this case, the solution is $t = 18$.

$$\left(\frac{1-\theta}{1+\theta}\right) = \left(\frac{1}{t+1}\right) \tag{7.2}$$

The Dervish field-trial data sets were tested using the overlapped hierarchical search with a block size of 27, bilinear interpolation and a mean resampling kernel. The reference region

update strategy was employed at both levels of the hierarchical pyramid. One important aspect is the rotation reference set, which removes the need to recalculate rotated reference regions that are already available from the previous frame. The only practical solution is to use the updated reference region to calculate new rotated reference regions and to leave the older regions already in the reference set unaltered. Assuming average motion occurs the oldest regions in the reference set will only be 3 or 4 frames old and this should not present significant problems. However, if the rotation were to slow for an extended period the reference set could quickly become more than 10 frames old. A possible solution to this would be to always recalculate at least $\frac{1}{3}$ (average motion) of the reference set and if less than average motion occurs use any remaining calculations to update any outdated reference regions within the set. However, this adds complexity to the system and has not at this stage been implemented. If aging reference sets proved a particular problem this method could be adopted.

The results are shown in Table 7.2 using the basic method outlined above and it can be seen that the DLS reference block update strategy has slightly mixed results. However, it should be noted that the accuracy degradation in the simpler sequences is considerably smaller than the accuracy improvement in the two grass sequences. Since the grass sequences represent a difficult but important area of operation they must be considered highly important.

Sequence	**no reference update**		**track–initiated DLS**	
	XY (pixels)	**θ *(deg)***	***XY (pixels)***	**θ *(deg)***
Concrete	2.5 (1.6)	1.1 (0.4)	2.4 (1.5)	1.4 (1.5)
LeavesStable	4.7 (4.9)	6.2 (14)	5.3 (8.7)	6.5 (11)
LeavesUnstable	7.5 (12)	4.1 (6.3)	7.6 (13)	4.1 (6.8)
GrassRuler	6.7 (34)	3.5 (11)	5.4 (17)	2.6 (4.8)
Grass	9.9 (57)	5.0 (17)	7.2 (21)	2.8 (4.6)

Table 7.2: *The Dervish field-trial data sets are tracked with and without DLS reference block updating and the accuracy is shown. The results are mixed but it can be seen that there is a significant positive effect on the grass sequences that have, so far, proved very difficult to track.*

A key question with reference block updating is the accuracy-efficiency trade off and it is therefore important to have optimized computational performance figures for the DLS reference block updating. The calculation requires two data loads, a subtraction and addition per pixel within the reference region. Using very similar techniques to the correlation similarity measure calculations, the two data loads can be performed in one cycle if the two data sources

are in separate memory banks. The calculation of the similarity measure required the current image data to be held in the Data-Memory bank and the reference image data to be held in the Program-Memory bank allowing the dual-data load to be performed. The task of optimizing the SHARC code is straightforward and the code is shown in Algorithm 7. The final number of clock cycles per pixel is 5 and the measured timing for a 27×27 pixel region is 3650 clock cycles. However, using the SIMD extensions it is possible to generate efficient code and the computation reduces to 1880 clock cycles. In comparison to the costs for interpolation and calculation of the similarity measure this is very small.

```
m5=m13;                                                  Set read increment DM
m12=2;                                                   And read increment PM
m0=m12;                                                  Set write increment
lcntr=364,        do(pc,_END-1)until lce;                start of loop
r13=dm(i2,m5),    r2=pm(i12,m12);                        Load A and B
F4=F2-F13;                                               Sub A and B
F12=F8*F4;                                               Weight (A-B)
F1=F13+F12;                                              Add W×(A-B)
dm(i2,m0)=r1;                                            Write result
_END:                                                    end of loop
```

Algorithm 7: *SHARC optimized routine for calculating the DLS reference block updating strategy using parallel data loads within the main routine. Data source A is the true reference block and data source B is the estimates.*

Therefore, given that reference block updating has been shown to provide increased stability and accuracy in general and that an increase in accuracy has been shown within the Dervish field-trial data sets, it would seem an advantage to include it within the correlation-based feature tracking. The total computational cost for both levels of the pyramid is 2100 clock cycles per feature per frame at a block size of 27 pixels. The accuracy improvement due to reference block updating is also tested within the passive navigation system, where the block lifespan varies considerably.

7.2.3 Feature Dissimilarity Measure

The KLT [87] uses a threshold on the SAD, calculated between the block at the new position of best match and the original reference block, to determine when a feature has become too dissimilar and needs to be discarded. However, the reference block updating is designed to allow changes in features, which would otherwise be discarded. The dissimilarity measure must, therefore, be adapted to take account of the reference block update strategy. Figure 7.3

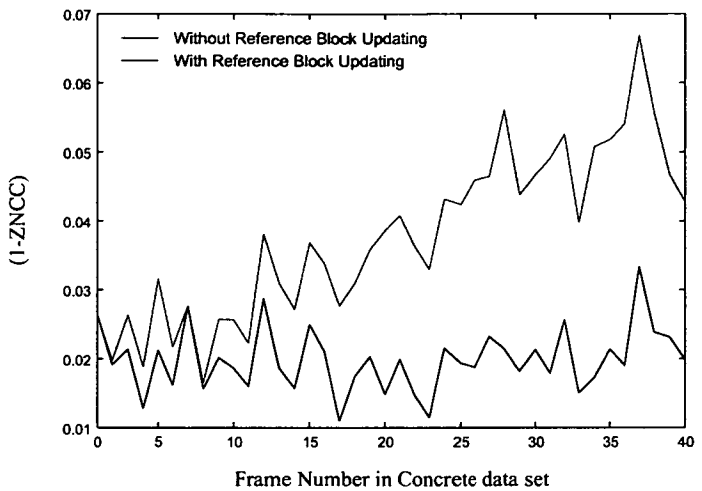


Figure 7.3: *Figure showing the minimum (1-ZNCC) when tracking with and without reference block updating.*

demonstrates this problem when tracking the same feature with and without reference block updating in the Concrete data sequence.

The simplest adaption would be to use a threshold on the similarity measure, calculated between the block at the new position of best match and the latest estimate for the reference block from the filter. In this case a feature is discarded if the match for a single frame is too dislike the filtered reference block for a single frame. However, a serious disadvantage to this technique is that there is no connection to the original reference block and no way of judging if the changes within the reference block accurately reflect changes within the target. This approach is essentially under-constrained and Figure 7.3 shows that the minimum correlation similarity measure under reference block updating tends to stay very steady, despite their being a significant variation in the tracking accuracy of each frame. This steady correlation measure can make it very difficult to identify lost features. Another minor problem with this technique, that also affects the original KLT dissimilarity measure, is that a single poor match results in the feature being discarded. In certain situations, such as where good features are scarce and problems, such as short-term occlusion, are possible this may present some problems. The simplest solution to this is to remove the feature from the passive navigation stage and to reinitialise it at the same place in the next frame.

A more complex adaption would be to use a dual-threshold on the similarity measure, where the difference from the filtered reference block and the original reference block. One way of viewing this would be that the overall motion (original reference block) and velocity (filtered

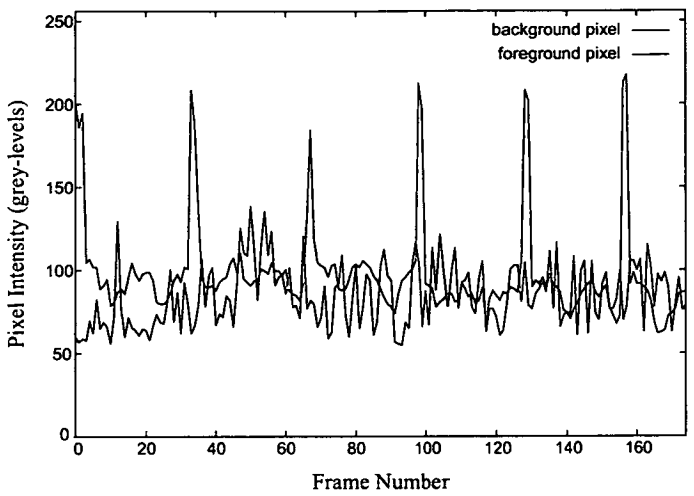


Figure 7.4: *Figure showing two pixels from the pedfollow block with one in the target (foreground) and one in the background*

reference block) are considered and if either exceed their threshold the feature is discarded. An empirical investigation would be necessary to determine suitable thresholds but identification for a particular application should be possible. However, this time the update strategy has become over-constrained. Given specific application knowledge it may be possible to select out of the two options already presented.

After some testing it became obvious that relating the current feature to the original reference block does negate the advantages of employing reference block updating. Therefore, the simple adaption of using the current reference block rather than the original reference block is preferred. Having reviewed the tracking of 20 different features (4 per field-trial data sequence) the smallest ZNCC selected as the best match was 0.79 and the largest ZNCC marking a tracking failure was 0.70. This provides a safety margin of 0.09 and suggests that the dissimilarity measure threshold should be set at 0.75.

The final method for dissimilarity is based entirely on the update filter over time and uses the difference between the predicted output and the actual output. If a pixel is not in the tracked target but is in the background the intensity variance will be much higher; Figure 7.4 shows two pixels from the Pedfollow block with one in the target (foreground) and one in the background. This filtered block variance can be of used in identifying foreground and background in target tracking application. An alpha-channel could be automatically generated and used to mask out background pixels allowing clearer identification of the target. Preliminary experiments on the Pedfollow data sets suggest that this method of background removal works well.

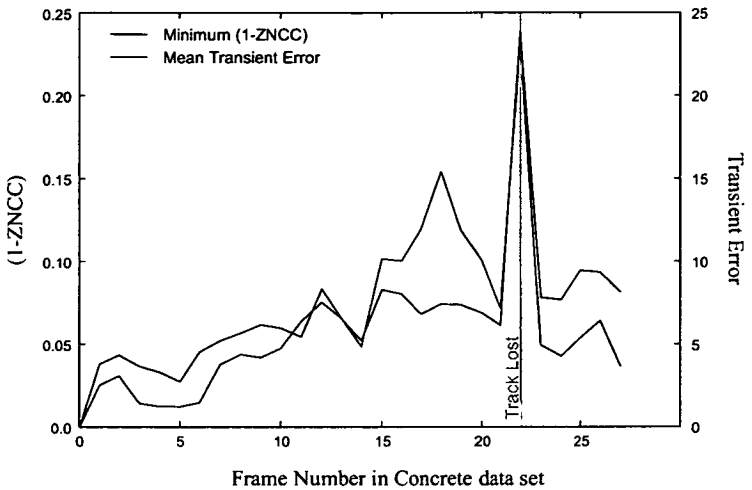


Figure 7.5: *The minimum (1-ZNCC) and average transient error are shown for a feature tracked through a short section of the Concrete data sequence. The feature being tracked is lost at frame 22, which is marked on the figure, and both dissimilarity measures clearly show a marked peak.*

In a downward looking vehicle navigation system, where all pixels are foreground, the intensity variance across the pixels will increase if the track is lost because essentially random blocks are input into the reference block update filter. However, instead of calculating the intensity variance, which requires a costly local variance per pixel it is possible to use the transient error ($\epsilon_t = (y_t - R_{t-1})$), which is already calculated for use within the reference block filter, and to take an average transient error within the reference block. This time if the average transient error varies between frames by more than a preset percentage the block can be discarded as too similar. In testing the average change was a 2 fold increase in average transient error. Therefore, the threshold is set at a 1.8 fold increase for a feature to be discarded.

Figure 7.5 shows minimum (1-ZNCC), used as the first dissimilarity measure, and the average transient error, used as the second dissimilarity measure, tracking over a short section of the Concrete data sequence. The feature being tracked is lost at frame 22, which is marked on the figure, and both dissimilarity measures clearly show a marked peak. However, the average transient error has a clearer peak with a greater difference between peak and off-peak response, making it simpler to identify and more robust. The 20 features used previously to identify dissimilarity thresholds were also examined in this manner and the results for the average transient error were found to be consistently clearer. Figure 7.5 actually regains the track and the results carry on for a few frames.

The two dissimilarity measures proposed above are useful within a completed feature track-

ing system and are, therefore, further tested within the completed passive navigation system in Section 7.4. This seems more appropriate than making a decision based on the relatively small number of features that have been examined. One final point is that robust versions of many dissimilarity measures have been proposed [113] but they generally adapt the method of calculation rather than the items being calculated on. With the introduction of reference block updating it is necessary to determine what objects require comparison; having done this it may be possible to provide a robust variant.

7.3 Passive Navigation

7.3.1 Overview

Passive navigation is a large subject and there are many techniques with variations within each. It is, therefore, not possible to test them all against each other for this given application. However, a recent survey by Tian *et al.* [93] suggested that the classic least-squares technique first presented by Bruss and Horn [92] consistently provided the most accurate results and the key to this is the assumption of rigid-body planar motion. This is a very strong assumption and failure of the assumption will lead to breakdown of the technique. However, the strength of the assumption produces the robustness and accuracy.

The assumption of planar rigid body for the Dervish landmine-Clearance vehicle is reasonable as images will be from a downward looking camera operating in relatively flat terrain; for example agricultural fields. The 3D motion model generally applied in passive navigation is a nine-parameter model [92], although only eight are used to define motion. However, in Chapter 6 the motion model for the correlation-based motion estimation was chosen as rigid-body motion and was restricted to rotation and translation. At this stage, it would seem inappropriate to allow a nine-parameter model for a rigid-body when it is not physically possible for the Dervish vehicle to undergo many of the possible permitted motions. The Dervish vehicle must remain in good contact with the ground at all times and it can reasonably be assumed that should the Dervish vehicle encounter any problems, such as falling down a steep slope, operation would be terminated and the sweep of that patch of ground restarted from a known position. Furthermore, the camera set-up is predetermined and the centre of rotation will be known *a priori*. Given the extensive knowledge about the camera set-up and vehicle motion, it is possible to make strong assumptions about the motion in an effort to simplify the the numerical minimization.

7.3.2 Linear Least-Squares Minimization

One key difference with this problem is that there is both translation and rotation available for every feature and the centre of rotation is known *a priori*. In contrast normally only a translation estimate would be known and the rotation would have to be derived from the passive navigation. The rigid-body motion model, presented in Section 6.3, is complex enough to handle the motion from the Dervish vehicle yet has only four parameters; it is defined as:

$$\mathbf{x}' = \kappa_{scale} \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \mathbf{x} + \mathbf{d} \tag{7.3}$$

where κ_{scale} is a scaling factor, θ is the angle of rotation and $\mathbf{d}$ is the displacement vector. However, the inclusion of trigonometric functions make the minimization non-linear and considerably harder. The affine motion model is also complex enough, however, the problem with it, as stated in Section 6.3, is that the parameters do not map directly into real-world motion. Allowing transformations that are not physically possible by Dervish during correct operation can make the minimization task considerably harder and subsequently the outlier rejection task. Essentially, by removing the trigonometric functions from the rigid-body motion model it introduces freedom that is unnecessary.

A simple way of handling this problem is to perform a minimization on the rotation estimates and to then un-rotate the motion estimates leaving only pure translation. The pure translation can then be subject to a separate minimization. In both cases the minimization problem is very simple. Table 7.3 shows this method tested on artificially generated motion estimates and it can be seen that performance remains high under increasing levels of Gaussian noise.

The method outlined above provides a simple and direct approach to the problem but relies heavily on accuracy of the rotation estimates. In particular it assumes that the error distribution is matched between rotation and translation; essentially that by independently identifying the rotation estimate you will subsequently be able to accurately identify the translation estimate.

If the rotation estimates were in any way suspect an alternative solution would be to solve for the following transformation:

Artificial Motion Estimate	**Artificial Noise (σ)**	***(X,Y) pixels***	**θ *deg***
Actual Position (2,5,12°)	0	(2.0, 5.0)	12.0°
	1	(2.0, 4.9)	11.9°
	3	(1.9, 5.4)	12.3°
	5	(1.8, 5.1)	11.5°
	10	(1.4, 4.1)	12.1°
	15	(5.6, 12.7)	3.6°
Actual Position (8,6,2°)	0	(8.0, 6.0)	2.0°
	1	(8.0, 6.0)	1.9°
	3	(6.1, 6.4)	1.9°
	5	(8.3, 6.3)	2.1°
	10	(6.9, 5.9)	1.6°
	15	(1.9, 2.1)	8.5°

Table 7.3: *An irregular grid of 50 artificially generated motion estimates is used to test the two-part minimization that splits rotation and translation. The results demonstrate excellent accurate under medium amounts of Gaussian noise but there are no data outliers within the artificial motion estimates.*

$$\mathbf{x}' = \kappa_{scale} \begin{bmatrix} a_1 & -a_2 \\ a_2 & a_1 \end{bmatrix} \mathbf{x} + \mathbf{d} \tag{7.4}$$

where a_1 and a_2 are independent variables. Although this does not represent θ within the system of equations it does provide a constraint towards the true rigid-body motion model. Applying a normal linear least-squares fit to this equation and solving the final system of equations with a Gaussian elimination produces very accurate results on synthetic data. Table 7.4 shows the results on synthetically generated motion estimates. The estimates for a_1 and a_2 agree to within two-degrees even under large amounts of Gaussian noise and have been averaged to produce the final result. A small draw back to this technique is that the final system of equations has a large number of zero's and full-pivoting within the least-squares method was required to solve on many occasions.

The two methods presented tend to have graceful degradation until a break-point after which they produce nonsensical results, such as translations of many millions of pixels. During testing

Artificial Motion Estimate	**Artificial Noise (σ)**	***XY (pixels)***	**θ *(deg)***
Actual Position (2,5,12°)	0	(2.0, 5.0)	12.1°
	1	(2.3, 4.9)	11.8°
	3	(2.4, 5.3)	10.5°
	5	(2.3, 4.5)	13.0°
	10	(2.9, 5.1)	7.8°
	15	(7.6, 16.2)	–2.6°
Actual Position (8,6,2°)	0	(7.8, 6.2)	2.3°
	1	(8.1, 5.4)	3.1°
	3	(8.1, 5.8)	3.3°
	5	(7.4, 7.6)	1.6°
	10	(7.2, 6.0)	6.3°
	15	(2.8, 0.1)	18.2°

Table 7.4: *An irregular grid of 50 artificially generated motion estimates is tested with a linear-least squares fit of the rigid-body motion model variant that removes trigonometric functions. Performance remains high under Gaussian noise but the artificial motion estimates contain no data outliers.*

it became clear that both the methods outlined above are very sensitive to the centre of rotation and to any outliers. The centre of rotation is no problem because it is known *a priori* and any damage to the camera would have to be repaired allowing recalibration. However, data outliers may represent a serious problem.

7.3.3 Robust Minimization

The two simple techniques presented for passive navigation have both been shown to be accurate under medium amounts of Gaussian noise. However, the sensitivity to data outliers is a serious problem. The total number of data points is relatively low, with a maximum of 48 features, and the chance of at least some losing track is high. While the feature dissimilarity measure will remove some features that have lost the track it is possible that others will reach the passive navigation stage.

Reviewing all the results for the correlation-based feature tracker from the previous chapter, it is obvious that the rotation estimate is very robust. In fact, the rotation estimate is consistently more accurate than the translation method. For this reason, the two-stage minimization is

considered suitable for the passive navigation stage. In this case a new formation for a robust solution can be proposed. The obvious way to find the rotation estimate would be to calculate the mean and while this is reasonably accurate, as shown in Table 7.3, it is influenced by data outliers. Instead, the more robust median can be employed to calculate the rotation through use of a least-trimmed squares technique. Here the median is calculated and all data points outwith a certain distance from the median are discarded. The trimmed data set is then minimized using standard least-squares techniques. In this two-stage minimization, the application is very similar to that of the X84 outlier rejection rule [113]. However, for computational efficiency a set number of features is used rather than a fixed number of median absolute deviations. In practice, under tests with a maximum of 48 features, it was found that on average 42 features entered the passive navigation stage and that selecting the best 34 features, based on the rotation estimate, proved most effective. This robust rotation can then be used to un-rotate the motion estimates to obtain pure translation. However, the robust rotation has already been used to discard some data outliers. Already, this produces a slightly smaller but more robust set of translation estimates.

The translation estimates must then be minimized and it would be possible to use a robust technique. However, testing showed that the combined use of a dissimilarity measure and robust rotation removed the need for robust minimization with little or no further accuracy to be gained. While this robust minimization, with its use of a median calculation, increases the overall cost, the systems that require minimization are very simple and are not comparable to the complex 8 or 9 parameter motion models normally employed. The next section shows the tests results for the two-stage minimization process.

7.3.4 Accuracy

The two-stage minimization process has been tested on four of the Dervish field-trial data sets. Unfortunately, the inclusion of a ruler within the GrassRuler sequences makes it unsuitable for the passive navigation testing. Table 7.5 shows the four data sequences tracked with the simple linear least-squares two-stage minimization and the robust variant. The accuracy of a single, near-optimal feature has also been shown for comparison. It can be seen that the robust minimization provides the highest accuracy. In particular, the maximum error and error variance for the translation improve considerably under the robust minimization. An indication of the search distance from the median that required to obtain the best 34 features is given. This

provides an indication of the variance between features in a sequence.

Sequence	Accuracy Type	Single Feature		Least–squares		Robust Minimization	
		(X,Y) pixels	θ *deg*	*(X,Y) pixels*	θ *deg*	*(X,Y) pixels*	θ *deg*
Concrete (±0.02 radians)	Max Error	6.0	6.8	5.6	6.2	5.4	3.4
	Mean Error	2.4	1.4	4.1	1.8	3.0	1.8
	Error Var	1.5	1.5	0.8	2.6	1.1	0.7
LeavesStable (±0.02 radians)	Max Error	11	15	9.1	17	7.2	19
	Mean Error	5.3	6.5	5.2	4.7	4.1	4.2
	Error Var	8.7	11	4.7	14	2.8	22
LeavesUnstable (±0.02 radians)	Max Error	14	10	11	7.3	9.9	6.4
	Mean Error	7.6	4.1	5.5	2.8	5.3	2.5
	Error Var	13	6.8	6.0	3.1	4.9	2.6
Grass (±0.06 radians)	Max Error	20	7.9	15	8.6	15	9.3
	Mean Error	7.2	2.8	6.7	4.7	6.7	5.4
	Error Var	21	4.6	18	5.3	18	4.7

Table 7.5: *The accuracy of the simple linear least-squares two-stage minimization and the robust variant are shown. The accuracy of a single, near-optimal feature has also been shown for comparison. The robust minimization demonstrates good accuracy with the maximum error and error variance, in particular, reducing significantly.*

At this stage of testing, with an increasing high level of accuracy in the tracking system, it is clear that the manual tracking is not exceptionally accurate. The mean error and error variance are acceptable for the application. The one remaining issue is the maximum error. Unfortunately, due lack of accurate ground truth, it is difficult to assess if the maximum error is acceptable. In particular, the accuracy of the grass sequence, which was very difficult to manually track, exhibits some unusual behaviour in the passive navigation tests. After visual examination of the feature data for the Grass, including region overlay on the image, it is impossible to tell if the manual track is correct. In fact, it would appear that the passive navigation is considerably more accurate than the manual track. Therefore, the exact performance figures for the Grass data sequence are not considered. The maximum error for the remaining sequences is 9.9-pixels, or 16mm, which is just outside the target, set in Chapter 2, of ±9 pixels. It is not clear at this stage if further accuracy would be required. In future work, a useful experiment would be to record a video sequence travelling over similarly difficult terrain (grass, mud, dirt) but to place small markers on the ground. The small markers could be forcibly ignored by the passive navigation but could be used to obtain accurate manual tracking. The GrassRuler se-

quence was considered for this purpose but the ruler proved too large a feature within the field of view.

The final parameter that must be tested is the dissimilarity measure. Two dissimilarity measures for use with reference block updating were proposed in Section 7.2. Table 7.6 shows the results of the two dissimilarity measures when tested within the robust passive navigation stage. It can be seen that neither dissimilarity measure have a significant effect on the tracking accuracy of any of the data sets. However, the transient error dissimilarity measure was identified in Section 7.2.3 as having slightly better performance and can be seen to have slightly better performance in this case. The use of a dissimilarity measure based on the transient error also provides a way of monitoring the track-initiation procedure from the reference block update filter should motion estimation errors cause it to lose target. For these reasons, the transient error dissimilarity measure is selected for use.

Sequence	**Accuracy Type**	**Robust Minimization**		**Minimum (1–ZNCC)**		**Transient Error**	
		(X,Y) pixels	*θ deg*	*(X,Y) pixels*	*θ deg*	*(X,Y) pixels*	*θ deg*
Concrete	Max Error	5.4	3.4	5.0	5.6	5.4	3.4
	Mean Error	3.0	1.8	2.9	1.7	3.0	1.8
	Error Var	1.1	0.7	0.9	1.3	1.1	0.7
LeavesStable	Max Error	7.2	19	7.2	19	7.2	19
	Mean Error	4.1	4.2	4.1	4.2	4.1	4.2
	Error Var	2.8	22	2.8	21	2.8	21
LeavesUnstable	Max Error	9.9	6.4	9.9	6.4	9.9	6.4
	Mean Error	5.3	2.5	5.3	2.5	5.3	2.5
	Error Var	4.9	2.6	4.9	2.6	4.9	2.6
Grass	Max Error	15	9.3	20	11	15	9.3
	Mean Error	6.7	5.4	8.4	6.3	6.8	5.4
	Error Var	18	4.7	41	6.6	20	4.7

Table 7.6: *The two dissimilarity measures are tested within the passive navigation stage using the thresholds identified in Section 7.2. The results of using only the robust minimization with no dissimilarity measure are shown for comparison. It can be seen that neither dissimilarity measure has a significant effect on the accuracy.*

In conclusion, a simple robust passive navigation technique has been proposed that represents a simplification of standard techniques. The design uses the *a priori* knowledge of system motion to restrict the minimization problem to a practical situation. It should be noted that

the author is aware of many alternatives to the proposed passive navigation technique. Many of the alternative passive navigation techniques could exhibit superior performance for this application. However, a simple but effective technique has been chosen that allows good testing of the underlying motion estimation techniques. A complete investigation into the suitability of different passive navigation techniques would provide an interesting but complex topic for further work.

7.4 Vision-based Navigation System Accuracy

7.4.1 Overview

The work up to this stage has focused on completing the design for the vision-based navigation system and testing of accuracy on the Dervish field-trial data sets. All parameters for the correlation-based feature tracking have been selected and confirmed through testing. The feature management and passive navigation systems have also been designed and tested. Testing has confirmed that over the short field-trial data sets the accuracy remains high even under problematic conditions, such as poor focus (Grass) and bouncing movement (LeavesUnstable). The error distribution for the tracking accuracy appears good with a very high mean error. The one problem is that the maximum error seen within the tracking was slightly higher than the target of ± 9-pixels. Even though this was only for one frame out of more than 45 seconds of footage it could still present a problem and will need to be examined in further field-trials.

The final area for testing is the tracking accuracy for the navigation system over the long term. A Dervish terrain sweep is likely to be in the region of 10 to 20 minutes and the testing so far has only on sequences of about 10 seconds. Unfortunately, the accuracy of manual tracking over longer periods is not particularly high. This is mainly due to the difficulty of identifying visual features to track and maintaining tracking of these features over an extended period. Therefore, the Dervish simulator is used to generate longer (>1 minute) sequences. The choice of ground image for use in the simulator is discussed in Section 2.3.4. Figure 7.6 shows the diamond-shaped path taken in the simulator sequence. The diagonal size of the diamond is approximately 1400 pixels, which using the current camera set-up would represent about 2.5m. The corner changes of direction are quite an important aspect of the motion estimation.

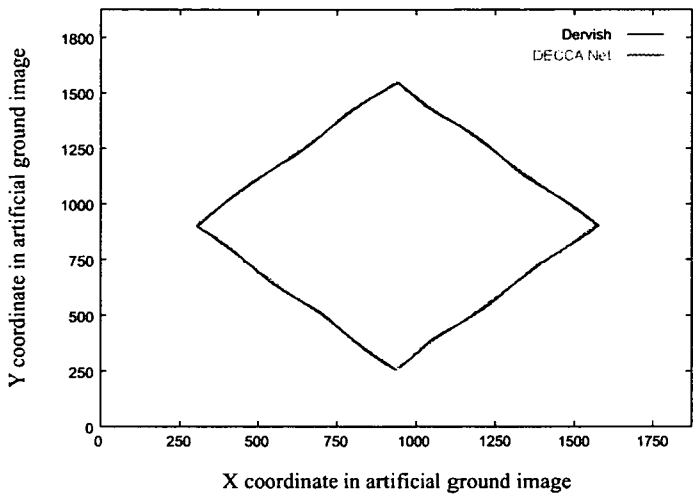


Figure 7.6: *The diamond-shaped path taken by the simulator image sequence. The size of the diamond diagonal is approximately 1400 pixels.*

7.4.2 Long-term Tracking Accuracy

Having completed the design for the vision-based navigation system it is now possible to test the accuracy of the system over an extended, though artificial, image sequence. The robust minimization is used for passive navigation and the transient error dissimilarity measure is employed and the artificial sequence covers 3 minutes and 20 seconds at 5 frames per second (1000 frames). Table 7.7 shows some results using the simulator sequence. The first column shows a sequence with the motion pattern seen in the Dervish field-trials but no artificial image noise and good contrast. The second column shows a very similar sequence with some artificial image noise added and the overall contrast lowered. The second sequence provides a reasonable representation of the sort of results seen in the field-trial sequences; the number of features, error distribution between features and overall accuracy being fairly well matched.

It can be seen from the results that the accuracy remains high throughout the two test sequences. There is no significant degradation in the performance of the system over an extended period of time. However, while the artificially generated image sequences provide a reasonable indication of accuracy variation over an extended sequence they do not give an indication of the absolute accuracy Therefore, it is very important that extensive field-trials take place with unhindered access to the vehicle for experimentation on frequent trials. It will also be necessary to have a completed Dervish control system to allow testing of the all systems integrated into the control loop.

An interesting point that emerged in this long-term testing is the importance of the accuracy

Sequence	Accuracy Type	1 minute		2 minutes		3 minutes 20 secs	
		(X,Y) pixels	*θ deg*	*(X,Y) pixels*	*θ deg*	*(X,Y) pixels*	*θ deg*
Simulator (Noise Free)	Max Error	7.7	2.8	9.2	5.0	10	9.0
	Mean Error	2.5	1.0	4.0	0.8	6.0	1.2
	Error Var	5.0	0.7	7.0	1.0	9.0	1.3
Simulator (Image Noise)	Max Error	15	4.9	18	8.6	19	14
	Mean Error	3.5	1.2	4.9	1.0	7.9	1.5
	Error Var	6.6	1.0	8.6	1.1	12	1.9

Table 7.7: *The tracking accuracy of the vision-based navigation system is shown for the 1000-frame Simulator sequences. A simple, noise free sequence and a more accurate sequence with artificial image noise are tested.*

of the rotation estimate. The passive navigation system relies on the rotation measurement to transform the individual motion estimates to isolate the translation. Increasing ground motion and overall rotation allows the amount of error in the translation to rapidly build up. Using the normal linear least squares passive navigation system the rotation accuracy degrades from 1.0 degrees mean error to 1.2 degrees mean error on the 300 frame sequence. While this may not seem highly significant it produces a degradation in translation accuracy from 2.5 pixels mean error to 6.0 pixels mean error, which is highly significant. This error in the rotation estimate continues to build throughout the sequences and after 2 minutes the track is essentially lost, even though the individual features are still tracking perfectly well.

One area that may be worth reviewing in future work is a final rotation-only search stage in the correlation-based feature tracking. It would be possible to perform more detailed search once the translation is identified, in a similar manner to the mid-level in the three-level search proposed in Chapter 6 but to have the search as the final level. Another option would be to perform a curve fitting procedure to the rotation because the rotation data, demonstrated in Figure 6.5, is very good around the correct translation. However, it would only be worth investigating these methods, which will increase the computational load, if the rotation accuracy proved a problem in further field-trials.

7.4.3 Accuracy Comparison

There are now several major motion techniques with many algorithm variations within each. To further complicate the problem, there are normally at least half a dozen parameters within

each algorithm. This makes any comparison of motion estimation algorithms, let alone passive navigation algorithms, very difficult. The comparisons will also have to be over a restricted data set, either through pre-set test sequences or through a specific application area, and will have to be on a restricted set of algorithms and parameters. To some extent this makes any attempt at a comprehensive comparison redundant due to the level of restrictions required to make the comparison feasible. Therefore, the main reason for providing an accuracy comparison is to demonstrate that standard alternative techniques do not perform significantly better or worse.

The two comparison techniques chosen are the Kanade-Lucas-Tomasi feature tracker and a hierarchical differential optical flow. These two techniques represent obvious alternatives, particularly the KLT tracker. The parameters for each of the motion estimation algorithm were investigated. The parameters used provide the best solution found but are not considered optimal. For both cases the rigid-body passive navigation structure, discussed in Section 7.3.2, was employed. Table 7.8 shows the results for the comparison.

Sequence	**Accuracy Type**	**Proposed Technique**		**KLT Tracker**		**Differential Opticflow**	
		(X,Y) pixels	θ *deg*	*(X,Y) pixels*	θ *deg*	*(X,Y) pixels*	θ *deg*
Concrete	Max Error	5.4	3.4	11	9.1	17	14
	Mean Error	3.0	1.8	6.3	5.8	9.3	11
	Error Var	1.1	0.7	2.3	4.5	4.7	11
LeavesStable	Max Error	7.2	19	15	25	22	32
	Mean Error	4.1	4.2	8.6	11	13	13
	Error Var	2.8	21	5.9	34	8.7	9.7
LeavesUnstable	Max Error	9.9	6.4	21	14	31	22
	Mean Error	5.3	2.5	11	5.3	16	14
	Error Var	4.9	2.6	10	8.5	15	13
Grass	Max Error	15	9.3	32	27	47	28
	Mean Error	6.8	5.4	14	11	21	17
	Error Var	20	4.7	42	14	62	15

Table 7.8: *A comparison of the proposed tracking technique against the Kanade-Lucas-Tomasi feature tracker and a hierarchical differential optical flow algorithm.*

The accuracy of the two alternative techniques is not as high as the proposed technique that has been carefully designed and developed in this thesis. Given the lack of detailed investigation into either of these two this is not a surprising result and better accuracy could almost certainly be achieved with both techniques given detailed application specific development. The per-

formance of the differential optical flow is heavily influenced by the poor image quality. The noise, vibration and poor focus, which effect all test data sets to a varying degree, make it very difficult for the numerical differentiation. This could probably be improved through use of an appropriate filter [159].

The performance of the KLT tracker is good considering that each feature is tracked in translation only. Unfortunately, this leads to many features being replaced very quickly. However, it would be possible to adapt the KLT motion model to allow translation and rotation. This would make the KLT tracker very similar to the proposed technique, with the key difference being the minimization technique. It may also provide an interesting accuracy-efficiency curve. However, the KLT tracker is very unlikely to increase the accuracy of the track and the proposed system is marginally accurate enough. Any saving in computation from the KLT would be irrelevant if the accuracy drops even slightly.

7.5 Vision-based Navigation System Performance

7.5.1 Bittware Hammerhead Implementation

The Bittware Hammerhead development board has four processors and it is important to make use of the combined processing power. Distributing a processing load between multiple processors is a complex task and many different solutions exist. The first stage of this application is to determine the exact nature of the problem. The main components of steady-state operation are:

1. I/O Data Transfer
2. Image pre-processing
3. Image Pyramid computation
4. Correlation-based tracking of multiple features
5. Homography of individual motion estimates

The most computational intensive component is the correlation-based tracking of multiple features. This will account for almost 90 percent of the total computation. The Dervish landmine-clearance vehicle also has some data processing required for wheel control and data telemetry.

The tasks of image pre-processing and homography of individual motion estimates are required to be performed after the image pyramid computation and cannot be parallelised. It is not known at this stage what volume of processing will be required for the Dervish control tasks. However, currently they are being performed on an 8-bit 1MHz micro-controller and unless considerable expansion takes place they will take a very small portion of the total processing power. The other factor that is not clear at this stage is if Dervish control system will accommodate a lag in the vision-based feature tracker.

One final problem that will effect any load distribution scheme is that the correlation-based feature tracking does not have a fixed computation time due to the rolling rotation reference set and the possible variation in computation which is 33 percent. Furthermore, the feature initialisation cost, due to the calculation of the complete rotation set is significantly higher. It is also important to remember that, assuming all features are tracking accurately, any variation in computation will be across the entire feature set because the rolling rotation reference set is the only place that computation varies and the same rotation will be present in every feature. Therefore, there are two options: place an upper threshold on the computational cost or reduce the total number of features tracked to allow maximum rotation between frames. In this case, 1 million clock cycle blocks are proposed for each feature to be tracked and this allows enough time for the the maximum rotation set to be calculated. While this may be slightly wasteful it does allow some spare computation in case of delays or unexpected interrupts that would otherwise cause significant problems. The image pre-processing and feature selection have been allowed 1 block while the passive navigation has been allowed two blocks (2 million clock cycles). These are based on timings of the implementation on the target hardware platform and they allow a reasonable amount of slack within the system. Two separate load distributions are now proposed below that attempt to provide two possible alternatives, given the unknown factors described above.

Having dealt with the aspects of steady state computation above the issue of feature selection must now be dealt with. Any other minor house keeping tasks could also be covered within this section. The vision-based short-range navigation system would not expect to be active until necessary. Therefore, upon activation there is the extra cost of full feature initialization. However, the full feature initialization can be performed in approximately $\frac{1}{4}$ second and this small delay should prove an acceptable initialization period for the Dervish to be stationary. A more important task is feature selection within continual operation. Here a new reference

rotation set must be generated and there is not enough computation time within the 1 million cycle block to do this. The approach to this is to allow a small amount of extra computation, outside of the normal correlation-based feature processing, to compute the extra rotation sets.

7.5.2 Multi-Processor Load Distribution

The most likely scenario is that any lag in the delivery of results from the vision-based feature tracker will cause problems but that the Dervish control systems will not require extensive processing. Therefore, functional decomposition of the 48 features onto the four processors provides the frame-work. The remaining tasks are placed immediately after image arrival and sit between the correlation-based tracking processes in terms of time. The advantage to this system is that every processor is working 100-percent of the time. However, a distinct disadvantage is that the Dervish control systems cannot gain access to the processors for long periods of the cycle outwith the designated slot after every frame delivery. While the entire system has some slack, designed to allow the Dervish control system some slack within which to work, particularly any interrupt driven processes, this time will not be enough for bulk processing. This method is shown in Figure 7.7 as the Compact system. The total processing time is 16 blocks and the lag time between frame delivery and navigation output is 18 blocks. On the current Bittware Hammerhead development platform this translates to a total processing time of 0.20 seconds, meeting the 5 frames per second, and a lag time of 0.225 seconds.

Alternatively, if the Dervish control system requires more extensive and more evenly distributed processing but can accommodate a longer lag time in the delivery of results from the vision-based feature tracker it is possible to arrange all the correlation-based feature tracking onto three processors and use the fourth to perform the remaining tasks. This functional-decomposition places 16 features on the first three processors with them performing no other tasks. This method is shown in Figure 7.7 as the Spare system. The major advantage to it is that one processor is normally free to perform Dervish control tasks. The total processing time is, once again, 16 blocks but the lag time increases to 23 blocks. On the current Bittware Hammerhead development platform this means that the 5 frames per second operation is met but the lag time is 0.288 seconds.

At this stage there is not enough information available on the Dervish control system requirements and no final decision can be made. Instead it will be necessary to examine the multi-processor configuration in more detail when all information is available. However, two suitable

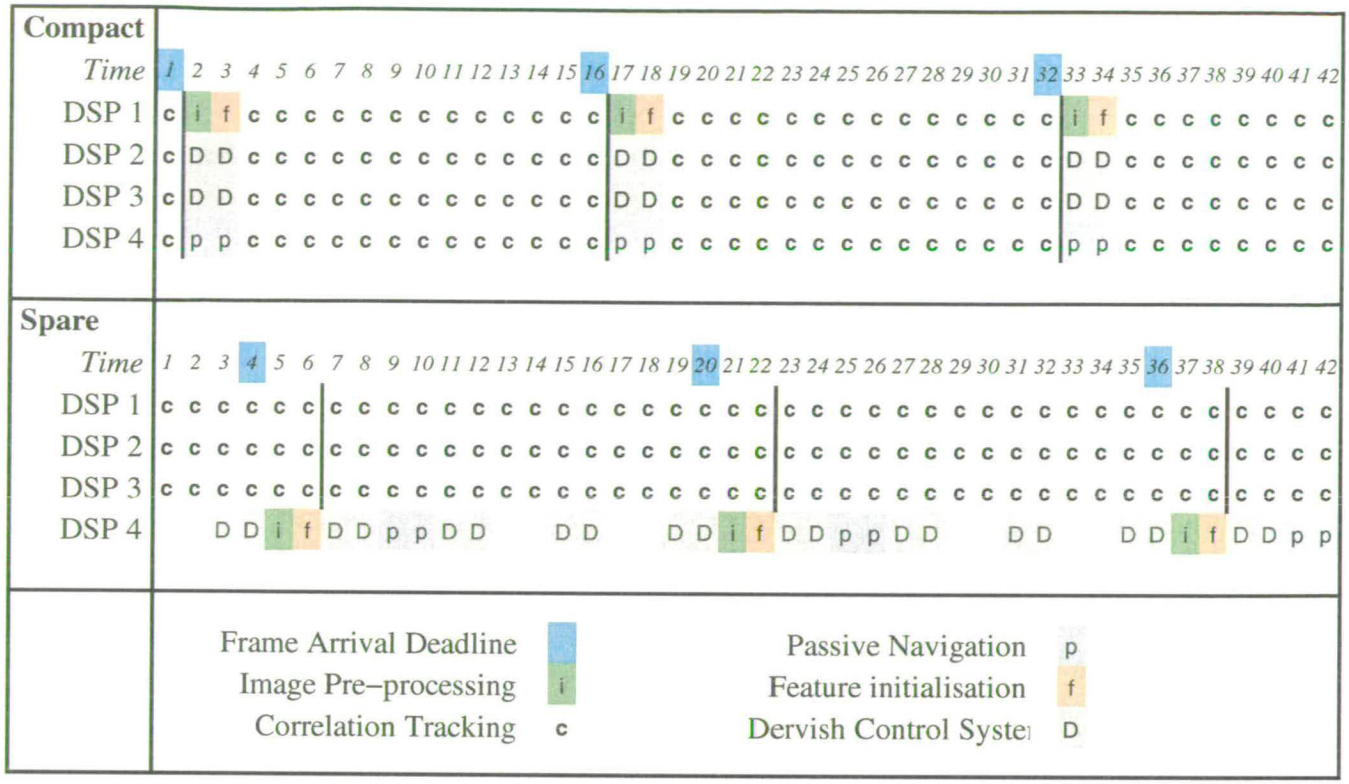


Figure 7.7: *Two different load distributions with their own advantages and disadvantages. In particular, the Compact method has very low lag time and high speed but has limited extra computation. In contrast, the Spare method has spare computation, evenly distributed but has a far higher lag time. The blocks of correlation-based feature tracking are synchronized at the end of each frame; this is shown by a vertical black bar on the processors concerned.*

arrangements have been demonstrated that would allow real-time motion estimation for the Dervish video-based navigation system. It should be noted that aiming for 100% processors usage is not advisable because any minor variation, such as a delay in data transfer or an unexpected processor interrupt, would cause a delay with no way of regaining time. In the suggested configurations the DSPs are approaching 95% capacity, which seems a reasonable balance.

7.5.3 Multi-Processor Datatransfer and Synchronization

The previous section discussed how to distribute the computational load between the different processors and proposed two different methods that have distinct advantages and disadvantages, depending on the circumstances. The next area that must be discussed is the data-transfer times and synchronization system. The starting synchronization is very simple because the image pre-processing is located on a single processor and all correlation-based feature tracking

processes must wait for pre-processing to complete. The two methods described above use functional-decomposition, placing a number of features on each of the processors, and, due to the large search ranges and rapid movement involved, the entire image must be transferred to each SHARC. The features move across the image too rapidly to make data-decomposition a real possibility. However, because the entire current image will have to be transferred to each SHARC, a bottle-neck will form and they will have to queue to transfer the data from the external memory bank to their internal memory bank. This is the only major data transfer operation required and the SHARC DMA controller can transfer memory, with no impact on the main processor, at almost 44 MBytes per second. This is enough to guarantee every SHARC will receive the data quickly and so will not have a large impact on the overall performance.

Data synchronization coming back is a slightly harder problem because multiple processors (3 or 4 depending on the load distribution scheme) have to provide data to the passive navigation process. The SHARC processors are already staggered in their computation from the image download but to completely avoid any similar bottle-neck problem the motion estimates for the individual features will be transferred in smaller groups. Depending on the final platform and its internal connections, the precise number may vary but initially it is set at groups of 5. This also means that, should a processor narrowly miss the target for computation of its features, only the final few features will be missing for the passive navigation process. In practice, the real-time system should meet requirements but this form of extra security is very useful in improving reliability, as the number of features varies naturally. A minimum number of features for correct operation was set in Section 7.4 and as long as the number of features does not drop below this operation could continue.

7.5.4 Analog Devices SHARC family processors

The development hardware platform was deliberately chosen to be the highest performance SHARC, the Hammerhead ADSP-21160N, to allow a good assessment of performance requirements. At this stage it is necessary to review the performance of the overall system and to try to determine whether a lower-cost platform could be successfully employed. The choice of DSP platform is limited to Analog Devices SHARC-based platforms because of the extensive development that has taken place with this architecture. The family of Analog Devices SHARC processors all share very high code compatibility allowing the majority of the development work to remain. There are three main SHARC processor families: SIMD SHARC,

SISD SHARC and the TigerSHARC but the first two of these provide the most obvious choice due to the almost perfect code compatibility.

Using the parameters defined above, a total of 48 blocks is being tracked across three processors. This is 16 blocks being tracked per processor and this currently takes almost all the processing power of the Bittware Hammerhead development platform. To allow enough computation space for all the other tasks, including Dervish control tasks, the use of multiple DSPs is helpful. However, since the SIMD SHARC processor is the latest on the market it commands a premium cost and the older SISD SHARC provides an almost identical structure but at a considerably lower cost. The major advantage of the SISD processor is that, apart from a few compiler directives detailing sections of code for SIMD optimization, all the code optimization will be identical and no reworking of the current system code would be necessary. Table 7.9 shows the predicated performance figures for the SISD SHARC. In fact many of the routines have actually been timed on the SISD SHARC and the prediction should be very accurate. It can be seen that a 16-SISD SHARC platform would be necessary to perform the vision-based navigation for Dervish. While a 16-SISD platform is available the extra complexity in utilising and even controlling 16 separate processors would be a significant overhead.

Processor	**Clock Speed**	**Computation Units**	**Computation Time (16 features)**	**Percentage Usage at 5fps**
SIMD Hammerhead	80MHz	2	16 000 000	95%
SISD SHARC	60MHz	1	42 000 000	350%

Table 7.9: *The computation time and percentage processor usage for two different processors is shown when tracking 16 features with the parameters selected in Chapter 6. The two SHARC processors compared are a very similar design but the SIMD SHARC contains a secondary computation unit and operates at a higher clock frequency.*

If a cheaper DSP were required there are high-performance Analog Devices fixed-point DSPs that have recently come on to the market such as the TigerSHARC. In this case it may be necessary to switch calculation of the similarity measure to the SAD to make use of their lower precision (8-bit or 16-bit) fixed-point SIMD architecture but all other code should remain identical. With the Analog Devices optimizing compilers it would only be necessary to hand optimize a small portion of the code, which would not be too significant a task. It is very difficult to predict performance for this processor family; in particular for the rotation and passive navigation code. Therefore, it would be necessary to conduct an investigation into a specific platform for this family of processors but it is thought that the TigerSHARC could

perform the vision-based navigation task for the Dervish landmine-clearance vehicle.

7.6 Summary

This chapter started off by discussing and selecting suitable feature management procedures for the Dervish vision-based navigation system. Novel reference block updating strategies, proposed in Chapter 4, were tested and found to be effective within this application. A novel variation on the dissimilarity measure was also presented, due to the introduction of the reference block update strategies. A suitable passive navigation system was also introduced and the accuracy and efficiency for the correlation-based feature tracking were examined separately. A key factor within the decision is whether to track more features less accurately or to track less features more accurately. This decision is heavily influenced by the number of good features that are available within the image sequence. A decision was made to track an evenly spaced grid of features, allowing the maximum number of features to be tracked. This still provides a practical maximum of about 48 features. The passive navigation problem is, in this case, simplified by *a priori* knowledge of the system behaviour and the estimation of both rotation and translation. Although the simple least-squares minimization performed well under Gaussian noise it was necessary to introduce a more robust median-based minimization to improve performance under data outliers.

It should be noted that the design of the correlation-based feature tracking system involves many complex and interlinked decisions that could be approached from alternative perspective. It has been shown that the proposed vision-based navigation system is capable of meeting the requirements set down by the Dervish. The final system design was tested and the motion estimation system compares favourably with alternative motion estimation algorithms.

The implementation on the target hardware platform was discussed and methods for multi-processor load distribution suggested. However, there are a number of Dervish sub-systems that are not fully complete and until all parameters are known it is difficult to make final decisions. At this stage it appears that the SIMD SHARC processors are required to provide the computational performance necessary. While the SISD SHARC processors do not provide enough performance, a cheaper alternative is the TigerSHARC multimedia processor, which have prices starting at ten dollars per processor. However, until all development of the Dervish control systems is closer to completion and more details are known, the development should

remain on the current Bittware Hammerhead platform. One important area that has not been addressed is the inclusion of a video source, such as digital camera or analog frame-grabber. There are a number of possible options for each DSP platforms but the DSP platforms use a variety of module connections such cPCI, PMC or proprietary connections such as SHARCPAC and BITSI. Another emerging option is a digital VGA camera with a USB 2.0 link, which could provide frame-rate data stream at an extremely low cost. All of these possible module connections have some form of camera / frame-grabber interface capable of operating at TV frame-rate but until the final hardware platform is chosen no decision can be made for the video source. It would also be necessary to consult with the Dervish design due to the necessity of protecting the camera equipment.

Chapter 8
Summary and Conclusions

8.1 Introduction

This thesis has investigated the problem of Motion Estimation in a vision-based navigation system for the Dervish landmine-clearance vehicle. It has concluded that a low-cost, real-time solution is possible for this *application* using the *algorithms* and *hardware* identified in this thesis.

The remainder of this chapter is divided as follows: Section 8.2 provides a summary of all the work presented in this thesis and identifies the main contributions. Section 8.3 draws conclusions from the work presented in this thesis, Section 8.4 presents some directions for future work and Section 8.5 gives some final remarks.

8.2 Summary

The theme of this thesis has been to examine motion estimation algorithms for the Dervish landmine-clearance project using an embedded hardware platform. Chapter 2 examined the Dervish landmine-clearance vehicle, providing information relevant to the project. Research and development on the Dervish are not complete and it is possible that this information may change in future. The Dervish was considered an interesting project for research mainly due to the highly unusual motion pattern and difficult working environment. A series of *application* requirements were derived in Chapter 2.

Chapter 8 provided a review of current motion estimation techniques for image processing. Having reviewed current literature and viewed the Dervish landmine-clearance vehicle in operation, correlation-based feature tracking was determined an appropriate technique and an outline motion estimation algorithm was proposed. A very important problem within the long-term tracking associated with vehicle navigation is updating of the reference region used to represent a feature. A common technique is to apply a dissimilarity measures to remove features that are

unlike the original target. However, a recent development applying a filter to slowly update a reference region allows more accurate tracking over extended image sequences. Therefore, Chapter 4 presents an investigation into reference block updating strategies. Computational performance and accuracy are examined with particular attention given to the accuracy under image noise. Under artificial image noise all reference block updating strategies are shown to have a relatively graceful degradation in performance before a cut-off when they fail to function correctly.

The optimized Kalman filters reference block update strategy is shown to have both the best computational performance and accuracy. A method for parameter selection is proposed but it is complex and subjective procedure. Instead a Discounted Least Squares filter is proposed that is shown to be equivalent to the optimized Kalman filter. However, a simple parameter selection procedure is given based on readily available filter information. Furthermore, a track-initiation procedure is proposed that helps avoid problems associated with the optimized Kalman filter. At this stage the *algorithm* requirements have been initially identified between Chapter 3 and Chapter 4.

Chapter 5 started by examining the *application* and *algorithm* requirements placed on the system. Using this information a review of possible hardware platforms was conducted and using these *hardware* requirements a Bittware Hammerhead SHARC DSP platform was chosen. A specific platform was selected and detailed information of the hardware is given in Chapter 5 and Appendix D. Some specific issues about the SHARC DSP were also presented to aid understanding of the optimization presented in Chapter 6 and Chapter 7.

Chapter 6 presented an investigation into the correlation-based feature tracking on the SHARC DSP, given the *application*, *algorithm* and *hardware* requirements for the Dervish landmine-clearance vehicle short-range vision-based navigation. Key areas of the correlation-based feature tracking were examined in detail for tracking accuracy and computational efficiency on data sets from the Dervish field-trials. Finally, the accuracy-efficiency trade-off was examined and suitable parameters chosen for the correlation-based feature tracker.

Chapter 7 described the implementation issues for the correlation-based feature tracker from Chapter 6 within a complete framework for vehicle navigation. Feature management was discussed and a novel technique for feature dissimilarity presented. The issue of reference block updating within ego-motion estimation was also examined and in this it was found to provide a

minor improvement in tracking accuracy. A suitable passive navigation technique was identified and used to determine accuracy of the final system design. This was compared with alternative motion estimation techniques and proved favourable. Finally, the Bittware Hammerhead development platform was used as a basis for examination of overall system performance and two multi-processor load distributions were proposed.

8.3 Summary of Findings

This thesis investigated a vision-based short-range navigation system for the Dervish landmine-clearance vehicle under constraints imposed by the *application*. It has been shown that with the *hardware* and *algorithm* a design can be achieved that meets both the real-time performance and tracking accuracy required by Dervish. However, the lack of unhindered field-trials presented a major obstacle and it would be necessary in any future work to have frequent trial runs on a fully functioning Dervish vehicle.

The correlation-based feature tracking techniques employed would seem to represent the best class of techniques for this problem and results have shown that the level of accuracy required is achieved, although without much room for manoeuvre. The algorithm adaptions detailed in Chapter 6 reduce the computation down to a reasonable level for the target hardware-platform. If any further reduction were necessary, a review of the KLT tracking algorithm may be appropriate. The target hardware platform would be suitable for integration within a final system but it may be possible to obtain a more compact, embedded PCI board that has the same components. The only drawback to the current hardware platform is the overall cost remains higher than preferred for the Dervish project. One way of addressing this would be to employ the Analog Devices TigerSHARC multimedia processor. The low-cost target market combined with the 16-bit SIMD operations may make it suitable. However, it would be necessary to investigate this further as code compatibility is lower due to the architectural differences. One pertinent point is that the floating-point DSP market is highly competitive and fast moving. An on-going reduction in the cost of the current hardware platform must be set against the time taken for any investigation of alternative platforms.

When designing a final system an important decision will be the resolution of objects within the field-of-view. It would be helpful to have a wider field-of-view because objects could be tracked over a longer period. However, it is the author's opinion that improving upon the

accuracy will be very difficult, given the application's operating situation. Therefore, it is important that any increase in the field-of-view does not decrease the resolution. In particular, if a higher resolution camera were employed with a wide-angle lens, image warping may be necessary before tracking and this could decrease the effective resolution of the image as well as increasing the computational load. Furthermore, the physical construction of the Dervish landmine-clearance vehicle makes obtaining a wider field-of-view very difficult. The camera setup used in field-trials provides a 256-level grey-scale image of 360×288 with each pixel on the ground representing 1.4mm vertically and 1.7mm horizontally. This would appear to be an appropriate camera set-up given the field-trial data seen at this stage.

The real-time frame rate target was set at 5 frames per second and this was achieved. Unless the Dervish control-system subsequently requires a higher frame rate there does not appear to be any need to increase the operating frequency. Although a system working at a higher frame rate would have a slightly smaller search area, the unpredictable motion will still require a very large search range and an increase in frame rate is unlikely to be matched by a reduction in computational cost. Therefore, extra computational performance would be required from the hardware platform.

8.4 Limitations

The work in this thesis is limited in a number of ways that may be extended in future work. The major limitation on the work in this thesis is the lack of test data sets with good ground truth. The best option would probably have been to place small markers on the ground to allow easier manual tracking of the image sequences. If the markers are small enough then the feature management process could have avoided features around the markers so that little effect was seen on the test results. Similarly, the work could have been improved by capturing some field-trial sequences at different resolutions. Although the video sequences were rescaled to a chosen image size, it would have been interesting to have a different field-of-view within some of the field-trial video sequences. Another limitation, due to lack of long-term ground sequences, is the lack of failure detection. The short field-trial sequences are all tracked very succesfully with the proposed system. However, in extensive field-trials it is likely that the system will fail under certain circumstances and it is very important that failure is identified.

One other area were the work is limited is the testing of alternative motion estimation tech-

niques. The motion estimation techniques tested within Section 7.4.3 do not perform as accurately on the field-trial data sets. Given further investigation into the parameters and preprocessing, an accuracy-efficiency investigation into other possible motion estimation algorithms, particularly the KLT variants, would provide a very interesting piece of work.

Finally, the passive navigation technique proposed is simple and effective while making good use of translation and rotation estimates. However, there are a large number of passive navigation techniques in the literature and a large number of robust strategies. The robust strategies may be able to make better use of the individual motion estimates to produce an improved tracking accuracy. The investigation into more complex passive navigation strategies would be major piece of research, complicated further by options within the motion estimation technique.

8.5 Future Work

The work in this thesis could be extended in the future in a number of ways. Firstly, Chapter 4 presents some work on reference block update strategies using confidence measures. This work was limited by the lack of reliable confidence measure but further investigation may lead to a suitable confidence measure. In general, the reference block update strategies presented in this thesis are likely to be of use in many correlation-based feature-tracking applications. Especially, where the target object is undergoing slow transformations that are difficult to model, such as a person walking.

Secondly, the work in thesis was based entirely upon the state of the Dervish landmine-clearance project between October 1999 and January 2001. During this period extensive field-trials were taking place for the development of mechanical control system and the DECCA navigation system. Due to the ongoing developmental work it would be necessary to re-examine the project in light of any changes or developments. In particular, the increasing accuracy of GPS systems may suggest a switch away from the DECCA navigation system. It is unlikely that a cheap, commercial GPS system would be able to achieve sub-centimeter accuracy in the near future, suggesting that a vision-based short-range navigation system would still be required. Future developments on the Dervish landmine-clearance vehicle do not effect the work currently presented in this thesis.

One area of Dervish development that did limit the scope of this thesis was lack of field-trials. Although significant field testing took place there was considerable time spent on the mechan-

ical aspects of Dervish. Video sequences taken were often of non-characteristic movement; for example, where either the Dervish mechanics or the DECCA navigation failed. While these situations represent an interesting set of operating conditions they almost all signify an operation abort and present limited data for testing of the vision-based navigation system. In the next stage of field-trials when the major Dervish systems are working reliably more time could be spent on development of vision-based navigation system. To allow true field-trial testing it would be necessary to mount the vision-based navigation system onto the Dervish through the embedded PC system. While this was not possible during the course of the work reported in this thesis due to the heavy constraints on the field-trials, every effort has been made to ensure that research during this thesis remains focused on producing the final on-board system.

A very interesting area for future work that was outside the scope of this thesis is the integration of the Dervish mapping system, DECCA navigation and short-range navigation system. Essentially, the system has multiple inputs, including the two navigation systems and the on-board electronic compass, and multiple outputs, including the Dervish ground mapping system.

8.6 Final Remarks

While motion estimation is a important and very active research area there is still much work to be done in allowing the accuracy seen in laboratory experiments to extend to real-world applications. The large majority of work that has taken place on real-world applications has been highly focused on a few target areas, such as corridor-based vehicles or autonomous road-based vehicles. The application to a real application presented issues that are not normally raised within the literature. In particular, it was necessary to conduct a careful review of accuracy and efficiency for the target application because incorrect selection of any of the many parameters, even by a small amount, led to widely varying tracking accuracy and computational performance. The inclusion of reference block updating is an important contribution to the field of correlation-based region tracking with application to many common tracking problems. Within this application, reference block updating provides increased robustness of tracking and extends the useful life of many features. This investigation concludes that a vision-based navigation system could be extremely valuable to the Dervish landmine-clearance vehicle project.

References

[1] Dervish Mine Clearance Ltd., "Dervish Website." www.dervish.org.

[2] S. H. Salter and C. N. G. Gibson, "Electronic navigation systems for the dervish and other mine-detonating and detecting vehicles," in *International Workship on Sustainable Humanitarian Demining*, pp. 1–17, Humanitarian Demining Information Centre, James Madison University, September 1997.

[3] S. H. Salter and C. N. G. Gibson, "Map-driven platforms for moving sensors in minefields," *Journal of Mine Action*, vol. Issue 3.2 (Mechanically Assisted Deminig), 1999.

[4] S. S. Beauchemin and J. L. Barron, "The computation of optic flow," *ACM Surveys*, vol. 27, pp. 433–467, September 1996.

[5] D. Touretzky, H. Wan, and A. Redish, "Neural representation of space in rats and robots," in *Computational Intelligence: Imitating Life* (J. M. Zurada and R. J. Marks, eds.), Proceedings of the symposium at the 1994 IEEE World Congress on Computational Intelligence, IEEE Press, 1994.

[6] J. H. Dripps, "Remote location of demining vehicles," in *Sustainable Humanitarian Demining: Trends, techniques and technologies*, pp. 167–171, Humanitarian Demining Information Centre at James Madison University, 1997.

[7] T. Mukai and N. Ohnishi, "Object shape and camera motion recovery using sensor fusion of a video camera and gyro senson," *Information Fusion*, vol. 1, pp. 45–53, 2000.

[8] B. P. Douglass, *Doing Hard Time: Developing Real-Time System with UML, Objects, Frameworks and Patterns*. Object Technology Series, Addison-Wesley.

[9] E. R. Fossum, "Cmos image sensors: Electronic camera-on-a-chip," *IEEE Transactions on Electron Devices*, vol. 44, pp. 1689–1698, October 1997.

[10] B. Jähne and H. Haußecker, eds., *Computer Vision and Applications: A guide for students and practitioners*. Academic Press, ISBN: 0123797772 ed., 2000.

[11] K. Price, "Multiframe feature-based motion analysis," in *Proceedings of Pattern Recognition 1990*, vol. 1, pp. 114–118, IEEE, 1990.

[12] J. L. Barron, D. J. Fleet, and S. S. Beauchemin, "Performance of optical flow techniques," *International Journal of Computer Vision*, vol. 12, no. 1, pp. 43–77, 1994.

[13] J. O. Limb and J. A. Murphy, "Measuring the speed of moving objects from television signals," *IEEE Transactions on Communications*, vol. 23, pp. 474–478, April 1975.

[14] A. N. Netravali and J. D. Robbins, "Motion compensated television coding," *Bell systems Technology Journal*, vol. 58, pp. 631–670, 1979.

[15] B. K. P. Horn and B. G. Schunck, "Determining optical flow," *Aritifical Intelligence*, vol. 17, no. 1-3, pp. 185–203, 1981.

[16] B. K. Horn, *Robot Vision*, ch. 12 - Motion Field and Optical Flow, pp. 278–297. The MIT Electrical Engineering and Computer Science Series, The MIT Press, McGraw-Hill Book Company, 1986.

[17] D. Ballard and C. Brown, *Computer Vision*, ch. 3.6 and 7.2-7.4, pp. 103–105 and 199–225. Prentice-Hall, Inc., 1982.

[18] J. J. Little and A. Verri, "Analysis of differential and matching methods for optical flow," in *IEEE 1989 Workshop on Visual Motion*, pp. 173–180, IEEE, March 1989.

[19] A. Singh and P. Allen, "Image-flow computation: An estimate-theoritc framework and a unified perspective," *Computer Vision, Graphics and Image Processing: Image Understanding*, vol. 56, pp. 152–177, September 1992.

[20] M. Mesbah, "Gradient-based optical flow: A critical review," in *Proceedings of the Fifth International Symposium on Signal Processing and its Applications*, (Brisbane, Australia), pp. 467–470, August 1999.

[21] I. Cohen and I. Herlin, "A motion computation and interpretation framework for oceanographic satellite images," in *Proceedings of the International Symposium on Computer Vision*, (Florida, USA), pp. 13–18, IEEE, November 1995.

[22] S. Uras, F. Girosi, A. Verri, and V. Torre, "A computational approach to motion perception," *Biological Cybernetics*, vol. 60, pp. 79–87, 1988.

[23] L. Ng and V. Solo, "Selecting the neighbourhood size, shape, weights and model order in optical flow estimation," in *Proceedings of International Conference on Image Processing*, vol. 1, (Vancouver, Canada), pp. 38–41, IEEE, September 2000.

[24] H. Liu, T.-H. Hong, M. Herman, T. Camus, and R. Chellappa, "Accuracy vs efficiency trade-offs in optical flow algorithms," *Computer Vision and Image Understanding*, vol. 72, pp. 271–286, 1998.

[25] H. Bülthoff, J. Little, and T. Poggio, "A parallel algorithm for real-time computation of optical flow," *Nature*, vol. 337, pp. 549–553, February 1989.

[26] R. C. Gonzalez and R. E. Woods, *Digital Image Processing*, pp. 582–586. Addison-Wesley, 1992. Basic BMA.

[27] H. Gharavi and M. Mills, "Blockmatching motion estimation algorithms - new results," *IEEE Transactions on Circuits and Systems*, vol. 37, no. 5, pp. 649–651, 1990.

[28] Y. Baek, H.-S. Oh, and H.-K. Lee, "Block-matching criterion for efficient vlsi implementation of motion estimation," *Electronic Letters*, vol. 32, pp. 1184–1185, 1996.

[29] G. B. Rath and A. Makur, "A fast matching criterion for vlsi implementation of block-based motion estimation," *Signal Processing*, vol. 73, pp. 297–301, 1999.

[30] W. Mendenhall, *Introduction to Probability and Statistics*. Wadsworth Publishing Company, 1953.

[31] D. N. Bhat and S. K. Nayar, "Ordinal measure for image correspondence," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 20, no. 4, pp. 415–423, 1998.

[32] Y.-K. Chen, Y.-T. Lin, and S. Y. Kung, "A feature tracking algorithm using neighbourhood relaxation with multi-candidate pre-screening," in *Proceedings of the International Conference on Image Processing*, vol. 2, (Lausanne, Switzerland), pp. 513–516, IEEE, September 1996.

[33] S. Birchfield and C. Tomasi, "A pixel dissimilarity measure that is insensitive to image sampling," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 20, pp. 401–406, April 1998.

[34] J. R. Jain and A. K. Jain, "Displacement measurement and its application in interframe image coding," *IEEE Transactions on Communications*, vol. 29, pp. 1799–1808, 1981.

[35] M. Ghanbari, "The cross-search algorithm for motion estimation," *IEEE Transactions on Communications*, vol. 38, pp. 950–953, 1990.

[36] T. Koga, K.Iinuma, A.Hirano, Y.Iijima, and T.Ishiguro, "Motion compensated interframe coding for video conferencing," in *National Telecommunications Conference*, (New Orleans, LA), pp. G5.3.1–5.3.5, November 1981.

[37] R. Li, B. Zeng, and M. L. Liou, "A new three-step search algorithm for block motion estimation," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 4, no. 4, pp. 438–442, 1994.

[38] L.-M. Po and W.-C. Ma, "A novel four-step search algorithm for fast block motion estimation," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 6, pp. 313–317, June 1996.

[39] H. Nisar and T.-S. Choi, "An advanced center biased search algorithm for motion estimation," in *IEEE Proceedings of the International Conference on Image Processing*, vol. 1, (Vancouver, Canada), pp. 292–295, IEEE, September 2000.

[40] J. N. Pan, Y. Q. Shi, and C. Q. Shu, "Correlation-feedback technique in optical flow determination," *IEEE Transactions on Image Processing*, vol. 7, pp. 1061–1067, July 1998.

[41] J.-C. Tsai, C.-H. Hseih, S.-K. Weng, and M.-F. Lai, "Block-matching motion estimation using correlation search algorithm," *Signal Processing: Image Communications*, vol. 13, pp. 119–133, 1998.

[42] W. Li and E. Salari, "Succesive elimnation algorithm for motion estimation," *IEEE transactions on Image Processing*, vol. 4, pp. 105–107, January 1995.

[43] C.-H. Lee and L.-H. Chen, "A fast motion estimation algorithm based on the block sum pyramid," *IEEE Transactions on Image Processing*, vol. 6, pp. 1587–1591, November 1997.

[44] T. Camus, "Real-time quantised opticflow," *Real-Time Imaging*, vol. 3, pp. 71–86, April 1997.

[45] F. Decroos, P. Schelkens, F. Stiens, J. Cornelis, and V. Christopoulos, "Motion monitoring and classification with center-biased motion estimation," in *Proceedings of the International Conference on Image Processing*, vol. 1, (Vancouver, Canada), pp. 872–875, IEEE, Septmeber 2000.

[46] H. Oh and H. Lee, "Adaptive adjustment of the search window for the block-matching algorithm with variable block size," *IEEE Transactions on Consumer Electronics*, vol. 44, pp. 659–666, 1998.

[47] M. H. Chan, Y. B. Yu, and A. G. Constantinides, "Variable size block matching motion compensation with applications to video coding," *IEE Proceedings*, vol. 137, Part 1, pp. 205–212, August 1990.

[48] J. Malo, F. J. Ferri, J. Albert, and J. M. Artigas, "Splitting criterion for hierarchical motion estimation based on perceptual coding," *Electronic Letters*, vol. 34, 1998.

[49] R. A. Packwood, M. K. Steliaros, and G. R. Martin, "Variable size block matching motion compensation for object-based video coding," in *Proceedings of the Sixth International Conference on Image Processing and Its Applications*, vol. 1, (Warwick University, Conventry, UK), pp. 56–60, IEE, July 1997.

[50] K.-C. H. Y.-L. C. W.-C. Siu, "Priority search technique for mpeg-4 motion estimation of arbitrarily shaped video object," in *Proceedings of the International Conference On Image Processing*, vol. 3, (Thessaloniki, Greece), pp. 644–647, IEEE, October 2001.

[51] G. J. Sullivan and R. L. Baker, "Motion compensation for video compression using control grid interpolation," *Proceedings of the 1991 IEEE International Conference on Acoustic Speech Signal Processing*, vol. M9.1, pp. 2713–2716, May 1991.

[52] V. Sefredis and M. Ghanbari, "General approach to block-matching motion estimation," *Optical Engineering*, vol. 32, no. 7, pp. 1464–1474, 1993.

[53] Y. Nakaya and H. Harashima, "Motion compensation based on spatial transformations," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 4, no. 3, pp. 339–356, 1994.

[54] F. Lopes and M. Ghanbari, "Improved motion estimation by spatial transformations and overlap," in *Proceedings of the International Conference on Image Processing*, vol. 3, (Chicago, IL, USA), pp. 623–627, IEEE, October 1998.

[55] F. Lopes and M. Ghanbari, "Hierarchical motion estimation with spatial transforms," in *Proceedings of International Conference on Image Processing*, vol. 1, (Vancouver, Canada), pp. 444–447, IEEE, September 2000.

[56] E. H. Adelson and J. R. Bergen, "Spatiotemporal energy models for the perception of motion," *Journal of the Optical Society of America*, vol. A2, pp. 284–299, 1985.

[57] M. Ogata and T. Sato, "Motion-detection model with two states: Spatiotemporal filtering and feature matching," *Journal of the Optical Society of America*, vol. 9, pp. 337–387, 1992.

[58] D. J. Fleet and A. D. Jepson, "Computation of component image velocity from local phase information," *International Journal of Computer Vision*, vol. 5, no. 1, pp. 77–104, 1990.

[59] U.-T. Koc and K. J. R. Liu, "Dct-based motion estimation," *IEEE Transactions on Image Processing*, vol. 7, pp. 948–965, July 1998.

[60] F. Glazer, *Hierarchical motion detection.* PhD thesis, University of Massachusetts, Amherst, MA, 1987.

[61] P. Anandan, "A computational framework and an algorithm for the measurement of visual motion," *International Journal of Computer Vision*, vol. 2, pp. 283–310, 1989.

[62] P. Golland and A. M. Bruckstein, "Motion from color," *Computer Vision and Image Understanding*, vol. 68, pp. 346–362, 1997.

[63] J. Magarey, A. Kokaram, and N. Kingsbury, "Optimal schemes for motion estimation on colour image sequences," in *Proceedings of the International Conference on Image Processing*, vol. 2, pp. 187–190, IEEE, 1997.

[64] J. Konrad and E. Dubois, "Use of colour information in bayesian estimation of 2-d motion," in *International Conference on Acoustics, Speech and Signal Processing*, vol. 4, pp. 2205–2208, IEEE, 1990.

[65] J. Wei and Z.-N. Li, "Motion compensation in colour video with illumination variations," in *Proceedings of the International Conference on Image Processing*, vol. 3, pp. 614–617, 1997.

[66] A. Koschan, V. Rodehorst, and K. Spiller, "Color stereo vision using hierarchical block matching and active color illumination," in *Proceedings of the 13th International Conference on Pattern Recognition*, vol. 1, pp. 835–839, 1996.

[67] R. Mecke, A. A. Hamadi, and B. Michaelis, "A robust method for block-based motion estimation in rgb-image sequences," in *Proceedings of the 14th International Conference on Pattern Recognition*, vol. 1, pp. 663–667, 1998.

[68] J. M. Sanchiz and F. Pla, "Feature correspondence and motion recovery in vehicle planar navigation," *Pattern Recognition*, vol. 32, pp. 1961–1977, December 1999.

[69] G. Calvagno, F. Fantozzi, and R. Rinaldo, "Feature based global and local motion estimation for videoconferencing sequences," in *Proceedings of the International Conference on Image Processing*, vol. 3, (Thessaloniki, Greece), pp. 102–105, IEEE, October 2001.

[70] L. G. Brown, "A survey of image registration techniques," *ACM Computing Surveys*, vol. 24, pp. 325–376, December 1992.

[71] Q. Zheng and R. Chellapa, "A computational vision approach to image registration," *IEEE Transactions on Image Processing*, vol. 2, pp. 311–326, July 1993.

[72] Y. Liu and T. S. Huang, "Three dimensional motion determination from real scene images using straight line correspondences," *Pattern Recognition*, vol. 25, pp. 617–639, June 1992.

[73] I. K. Sethi and R. Jain, "Finding trajectories of feature points in monocular image sequences," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 9, no. 1, pp. 56–73, 1987.

[74] A. Giachetti, M. Campani, and V. Torre, "The use of optical flow for road navigation," *IEEE Transactions on Robotics and Automation*, vol. 14, pp. 34–48, February 1998.

[75] J. F. S. Yau and N. D. Duffy, "A feature tracking method for motion parameter estimation in a model-based coding application," in *Proceedings of the Third International Conference on Image Processing and its Applications*, pp. 531–535, 1989.

[76] B. Southall, T. Hague, J. A. Marchant, and B. F. Buxton, "Vision-aided outdoor navigation of an autonomous horticultural vehicle," *Lecture Notes in Computer Science*, vol. 1542, pp. 37–50, 1999.

[77] G. Sela and M. D. Levine, "Real-time attention for robotic vision," *Real-Time Imaging*, vol. 3, pp. 173–194, 1997.

[78] X. Feng and P. Perona, "Real time motion detection system and scene segmentation," CDS Technical Report CIT-CDS-98-004, California Institure of Technology, Pasadena, CA 91125, March 1998.

[79] J. Shi and C. Tomasi, "Good features to track," in *Proceedings of the International Conference on Computer Vision and Pattern Recognition*, pp. 593–600, IEEE, 1994.

[80] Y. S. Abu-Mostafa and D. Psaltis, "Recognition aspects of moment invariants," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 6, no. 6, pp. 698–760, 1984.

[81] K. Arbter, W. E. Snyder, H. Burkhardt, and G. Hirzinger, "Application of affine-invariant fourier descriptors to the recognition of 3-d objects," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 12, pp. 640–647, July 1990.

[82] J. S. Park and J. H. Han, "Estimating optical flow by tracking contours," *Pattern Recognition Letters*, vol. 18, pp. 641–648, 1998.

[83] L. Chen, Z. Lu, E. K. Teoh, and Z. Xue, "A novel affine invariant feature set and its application in motion estimation," in *Proceedings of the International Conference on Image Processing*, vol. 1, (Vancouver, Canada), pp. 281–284, IEEE, September 2000.

[84] C. J. Veenman, M. J. T. Reinders, and E. Backer, "Resolving motion correspondence for densely moving points," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 23, pp. 54–72, January 2001.

[85] I. J. Cox and S. L. Hingorani, "An efficient implementation of reid's multiple hypothesis tracking algorithm and its evaluation for the purpose of visual tracking," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 18, pp. 138–150, February 1996.

[86] V. S. S. Hwang, "Tracking feature points in time-varying images using an opportunistic selection algorithm," *Pattern Recognition*, vol. 23, no. 3, pp. 247–256, 1989.

[87] C. Tomasi and T. Kanade, "Detection and tracking of point features," Tech. Rep. CMU-CS-91-132, School of Computer Science, Carnegie Mello University, Pittsburgh, PA 15213, April 1991.

[88] T. Tommasini, A. Fusiello, E. Trucco, and V. Roberto, "Making good features to track better," in *Proceedings of the International Conference on Computer Vision and Pattern Recognition*, pp. 178–183, IEEE, June 1998.

[89] J. Weng, N. Ahuja, and T. S. Huang, "Optimal motion and structure estimation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 15, pp. 864–884, September 1993.

[90] C. Morimoto and R. Chellapa, "Fast electronic digital image stabilization for off-road navigation," *Real-Time Imaging*, vol. 2, pp. 285–296, 1996.

[91] R. C. Luo and H. Potlapalli, "Fractal based outdoor landmark recognition system for the navigation of a mobile robot," in *Proceedings of the International Conference on Robotics and Automation*, vol. 3, pp. 1973–1978, IEEE, May 1994.

[92] A. R. Bruss and B. K. P. Horn, "Passive navigation," *Computer Vision, Graphics and Image Processing*, vol. 21, pp. 3–20, 1983.

[93] T. Y. Tian, C. Tomasi, and D. J. Heeger, "Comparison of approaches to egomotion computation," in *Proceedings of Computer Vision and Pattern Recognition*, (San Francisco, CA), pp. 315–320, IEEE, June 1996.

[94] D. J. Heeger and A. D. Jeepson, "Subspace methods for recovering rigid motion i: Algorithm and implementation," *International Journal of Computer Vision*, vol. 7, pp. 95–117, February 1992.

[95] C. Tomasi and J. Shi, "Direction of heading from image deformations," in *Proceedings of the International Conference on Computer Vision and Pattern Recognition*, (New York, USA), pp. 422–427, IEEE, June 1993.

[96] K. Prazdny, "Egomotion and relative depth map," *Biological Cybernetics*, vol. 36, pp. 87–102, 1980.

[97] K. Kanatani, "3-d interpretation of optical flow by renormalization," *International Journal of Computer Vision*, vol. 11, pp. 267–282, March 1993.

[98] B. K. Horn, *Robot Vision*, ch. 17 - Passive Navigation and Structure from Motion, pp. 400–483. The MIT Electrical Engineering and Computer Science Series, The MIT Press, McGraw-Hill Book Company, 1986.

[99] S. Negahdaripour and B. K. P. Horn, "Direct passive navigation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 9, no. 1, pp. 168–176, 1987.

[100] S. Negahdaripour and B. K. P. Horn, "Direct passive navigation," Tech. Rep. A.I. Memo No. 821, Aritifcial Intelligence Laboratory, Massachusetts Institute of Technology, February 1985.

[101] S. Negahdaripour and B. K. P. Horn, "Direct passive navigation: Analytical solution for planes," Tech. Rep. A.I Memo No. 863, Aritifcial Intelligence Laboratory, Massachusetts Institute of Technology, August 1985.

[102] S. Negahdaripour and C. H. Yu, "Robust recovery of motion: Effects of surface orientation and field of view," in *Proceedings of the International Conference on Computer Vision and Pattern Recognition*, (Ann Arbor, MI), pp. 404–410, IEEE, June 1988.

[103] A. Branca, G. Cicirelli, E. Stella, and A. Distante, "Mobile vehicle's egomotion estimation from time varying image sequences," in *Proceedings of the International Conference on Robotics and Automation*, (Alberquerque, New Mexico), pp. 1886–1891, IEEE, April 1998.

[104] A. Branca, E. Stella, and A. Distante, "Passive navigation using egomotion estimates," *Image and Vision Computing*, vol. 18, pp. 833–841, July 2000.

[105] C. Silva and J. Santos-Victor, "Direct egomotion estimation," in *Proceedings of the International Conference on Pattern Recognition*, vol. 1, (Vienna, Austria), pp. 702–706, IEEE, August 1996.

[106] P. J. Rousseeuw and A. M. Leroy, *Robust Regression and Outlier Detection*. Wiley Series in Probability and Mathematical Statistics, Wiley, 1987.

[107] T. Zhang and C. Tomasi, "Fast, robust and consistent camera motion estimation," in *Proceedings of the International Conference on Computer Vision and Pattern Recognition*, vol. 1, (Fort Collins, CO), pp. 164–170, IEEE Computer Society, June 1999.

[108] A. Bab-Hadiashar and D. Suter, "Robust optic flow estimation using least median squares," in *Proceedings of the International Conference on Image Processing*, vol. 1, (Lausanne, Switzerland), pp. 513–516, IEEE, September 1996.

[109] J.-M. Odobez and P. Bouthemy, "Robust multiresolution estimation of parametric motion models," *Journal of Visual Communication and Image Representation*, vol. 6, pp. 348–365, December 1995.

[110] C. V. Stewart, "Bias in robust estimation caused by discontinuities and multiple structures," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 19, pp. 818–833, August 1997.

[111] F. Spindler and P. Bouthemy, "Real-time estimation of dominant motions in underwater video images for dynamic positioning," in *Proceedings of the International Conference on Robitcs and Automation*, vol. 1, (Leuven, Belgium), pp. 1063–1068, IEEE, May 1998.

[112] A. Smolic and J.-R. Ohm, "Robust global motion estimation using a simplified m-estimator approach," in *Proceedings of the International Conference on Image Processing*, vol. 1, (Vancouver, Canada), pp. 247–250, IEEE, September 2000.

[113] T. Tommasini, A. Fusiello, and V. Roberto, "Robust feature tracking in underwater video sequences," in *Proceedings of the International Conference on OCEANS*, vol. 1, (Nice, France), pp. 46–50, IEEE, September 1998.

[114] E. Sayrol, "Modelling the displaced frame difference as an alpha state distribution," in *Proceedings of the International Conference on Image Processing*, vol. 2, (Santa Barbara, CA), pp. 195–198, IEEE, October 1997.

[115] J. V. Beck and K. J. Arnold, *Parameter Estimation in Engineering and Science*. Wiley Series in Probability and Mathematical Sciences, Wiley, 1977.

[116] D.-G. Sim and R.-H. Park, "Robust reweighted map motion estimation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 20, pp. 353–365, April 1998.

[117] E. Brookner, *Tracking and Kalman Filtering Made Easy*. Wiley-Interscience, 1998.

[118] N. K. Papanikolopoulos, P. K. Khosla, and T. Kanade, "Visual tracking of a moving target by a camera mounted on a robot: A combination of control and vision," *IEEE Transactions on Robotics and Automation*, vol. 9, pp. 14–35, February 1993.

[119] Y. Petillot, I. T. Ruiz, and D. M. Lane, "Underwater vehicle obstacle avoidance and path planning using a multi-beam forward looking sonar," *IEEE Journal of Oceanic Engineering*, vol. 26, pp. 240–251, April 2001.

[120] K. Nickels and S. Hutchinson, "Measurement error estimation for feature tracking," in *Proceedings of the International Conference on Robotics and Automation*, (Detroit, MI (USA)), pp. 3230–3235, IEEE, May 1999.

[121] G. N. DeSouza and A. C. Kak, "Vision for mobile robot navigation: A survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, pp. 237–266, February 2002.

[122] R. Mandelbaum, M. Hansen, P. Burt, and S. Baten, "Vision for autonomous mobility: Image processing on the vfa-200," in *Proceedings of the International Symposium on Intelligent Control (ISIC), Computational Intelligence in Robotics and Automation (CIRA), Intelligent Systems and Semiotics (ISAS)*, (Gaithersburg, MD (USA)), pp. 671–676, IEEE, September 1998.

[123] W. Zheng, I. Ahmad, and M.-L. Liou, "Adaptive motion search with elastic diamonf for mpeg-4 video coding," in *Proceedings of the International Conference on Image Processing*, vol. 1, (Thessaloniki, Greece), pp. 377–380, IEEE, October 2001.

[124] A. M. Peacock, S. Matsunaga, D.Renshaw, J. Hannah, and A. Murray, "Reference block updating when tracking with block matching algorithm," *Electronic Letters*, vol. 36, pp. 309–310, February 2000.

[125] J. Kittler, M. Hatef, R. P. W. Duin, and J. Matas, "On combining classifiers," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 20, pp. 226–239, 1998.

[126] H. Tian, B. Fowler, and A. E. Gamal, "Analysis of temporal noise in cmos photodiode active pixel sensor," *IEEE Journal of Solid State Circuits*, vol. 36, pp. 92–101, January 2001.

[127] K. Rank, M. Lendl, and R. Unbehauen, "Estimation of image noise variance," *IEE Proceedings on Vision, Image and Signal Processing*, vol. 146, April 1999.

[128] S. W. Smith, *The Scientists and Engineers Guide to Digital Signal Processing.* San Diego, California: California Technical Publishing, second edition ed., 1997.

[129] R. B. Blackman, *Data Smoothing and Prediction.* Addison-Wesley, 1965.

[130] N. Morrison, *Introduction to Sequential Smoothing and Prediction.* McGraw-Hill, 1969.

[131] X.-D. Zhang and C.-Y. Tsui, "An efficient and reconfigurable vlsi architecture for different block matching motion estimation algorithms," in *Proceedings of the International Conference on Acoustics, Speech and Signal Processing*, vol. 1, (Munich, Germany), pp. 603–606, IEEE, April 1997.

[132] Xilinx, "Xilinx Programmable Logic Website." www.xilinx.com.

[133] Xilinx, "Xilinx Virtex II DSP Core Website." www.xilinx.com/xlnx/xil_prodcat_ landingpage.jsp?title=Xilinx+DSP.

[134] A. Ryszko and K. Wiatr, "An assesment of fpga suitability for implementation of real-time motion estimation," in *Proceedings of the European Symposium on Digital System Design*, (Warsaw, Poland), pp. 364–367, IEEE, September 2001.

[135] S. Furber, *ARM System Architecture.* Addison-Wesley, 1996.

[136] M. A. Turk, D. G. Morgenthaler, K. D. Gremban, and M. Marra, "Vits - a vision system for autonomous land vehicle navigation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 10, pp. 342–361, May 1988.

[137] Philips, "Philips Trimedia Processors." www.semiconductors.philips.com/trimedia/.

[138] Coreco, "Coreco Imaging Website." www.coreco.com.

[139] T. Technology, "Datacube inc. homepage." www.datacube.com, 2002.

[140] I. Main, "Which Floating-Point DSP Offers Highest Performance," tech. rep., Spectrum Signal Processing, 1999.

[141] "Sharc-based signalmaster technical reference," tech. rep., LYR Signal Processing, 1999.

[142] Bittware, "Bittware Embedded DSP SHARC Solutions." www.bittware.com.

[143] Bittware, "Bittware SHARCPAC Website." www.bittware.com/products/IOmezzanines/SHARCPAC.stm.

[144] "ADSP-21160 Versus Texas Instruments TMS320C6X," tech. rep., Analog Devices, 1998.

[145] Xilinx, *XC4000E and XC4000X Series FPGA (Datasheet)*, 1999.

[146] T. Camus, "Real-time quantized optical flow," in *Proceedings of the International Conference on Computer Architectures for Machine Perception*, (Como, Italy), pp. 126–131, IEEE, September 1995.

[147] C. Haworth, A. M. Peacock, and D. Renshaw, "Performance of reference block updating techniques when tracking with the block matching algorithm," in *Proceedings of the International Conference on Image Processing*, vol. 1, (Thessaloniki, Greece), pp. 365–368, IEEE, October 2001.

[148] M. Chen, L. Chen, K. Cheng, and M. Chen, "Efficient hybrid tree/linear array architectures for block-matching motion estimation algorithms," *IEE Vision, Image and Signal Processing*, vol. 143, pp. 217–222, August 1996.

[149] A. Giachetti, "Matching techniques to compute image motion," *Image and Vision Computing*, vol. 18, pp. 247–260, 2000.

[150] W. Press, S. Teukolsky, W. Vetterling, and B. Flannery, *Numerical Recipes in C, The Art of Scientific Computing*. Cambridge University Press, 2nd edition ed., 1994.

[151] D. Dai, T. Shih, and F. Chau, "Polynomial preserving algorithm for digital image interpolation," *Signal Processing*, vol. 67, pp. 109–121, 1998.

[152] M. D. Macleod, "Interpolation using the discrete sine transform with increased accuracy," *Electronics Letters*, vol. 30, pp. 479–480, March 1994.

[153] H. S. Hou and H. C. Andrews, "Cubic splines for image interpolation and digital filtering," *IEEE Transactions on Acoustic, Speech and Signal Processing*, vol. ASSP-26, pp. 508–517, December 1978.

[154] M. Unser, A. Aldroubi, and M. Eden, "B-spline signal processing: Part 1 - theory," *IEEE Transactions on Signal Processing*, vol. 41, no. 2, pp. 821–833, 1993.

[155] M. Unser, A. Aldroubi, and M. Eden, "B-spline signal processing: Part 2 - efficient design and applications," *IEEE Transactions on Signal Processing*, vol. 41, no. 2, pp. 834–848, 1993.

[156] L.-J. Wang, W.-S. Hsieh, T.-K. Troung, I. S. Reed, and T. C. Cheng, "A fast efficient computation of cubic-spline interpolation in image codec," *IEEE Transactions on Signal Processing*, vol. 49, no. 6, pp. 1189–1197, 2001.

[157] J. D. Foley, A. van Dam, S. K. Feiner, and J. F. Hughes, *Computer Graphics: Principles and Practice*. Addison-Wesley, 2nd edition ed., 1996.

[158] H. P. Moravec, "Towards automatic visual obstable avoidance," in *Proceedings of the International Joint Conference on Artificial Intelligence*, p. 584, IEEE, 1977.

[159] M. Elad, Y. Hel-Or, and P. Teo, "Optimal filters for gradient-based motion estimation," in *Proceedings of the IEEE Convention of the Electrical and Electronic Engineers in Israel*, (Tel-Aviv, Israel), pp. 195–197, IEEE, April 2000.

[160] A. M. Peacock, *Information Fusion for Improved Motion Estimation*. PhD thesis, Department of Electronics and Electrical Engineering, the University of Edinburgh, May 2001.

[161] B. D. Lucas and T. Kanade, "An iterative image registration technique with application to stereo vision," in *Proceedings of the International Joint Conference on Artificial Intelligence*, (Vancouver, Canada), pp. 674–679, 1981.

[162] *ADSP-2106X SHARC User's Manual*, 1996.

Appendix A
Publications

Refereed Journals and Conferences

C. Haworth and D. Renshaw, "Image-sequence Motion Estimation for the Dervish Mine-clearance Vehicle", in *IEE proceedings on Science, Measurement and Technology*, pp. 89-94, vol 148, issue 3, May 2001

C. Haworth, A. M. Peacock, and D. Renshaw, "Performance of reference block updating techniques when tracking with the block matching algorithm", in *Proceedings of the International Conference on Image Processing*, vol. 1, (Thessaloniki, Greece), pp. 365-368, IEEE, October 2001.

Other Work

C. Haworth, "Image-sequence Motion Estimation for Dervish landmine-clearance vehicle", in *PhDEE - The Postgraduate Journal of the Department of Electronics and Electrical Engineering*, the University of Edinburgh, June 2002.

C. Haworth, "Real-time computation of speed of movement in image sequences", *BEng Honours Project*, Department of Electronics and Electrical Engineering, the University of Edinburgh, (HSP 1421), May 1999.

Appendix B
Background Derivations

B.1 Derivation of Horn and Schunck's Iterative Solution

This section describes all the steps for the derivation of an iterative solution for the differential optical flow method proposed by Horn and Schunck [15]. All the approximations and derivations have been made explicit. This derivation is adapted from Peacock [160].

Horn and Schunck start with the assumption that the intensity of an imaged point remains constant over time and motion.

$$\frac{d\mathcal{I}}{dt} = 0 \tag{B.1}$$

where $\mathcal{I}(x, y, t)$ is the instantaneous intensity of a pixel (x, y) at time t. Using the Chain Rule for differentiation this can be expanded to give:

$$\mathcal{I}_m = \frac{\partial \mathcal{I}}{\partial x}u + \frac{\partial \mathcal{I}}{\partial y}v + \frac{\partial \mathcal{I}}{\partial t} \tag{B.2}$$

where $u = \frac{dx}{dt}$ and $v = \frac{dy}{dt}$ are the horizontal and vertical velocity components. The second and higher order terms become zero as $\delta t \rightarrow 0$. This is referred to as the Motion Gradient Constraint and is referred to as $\mathcal{I}_m$. Setting $\frac{\partial \mathcal{I}}{\partial x} = \mathcal{I}_x$, $\frac{\partial \mathcal{I}}{\partial y} = \mathcal{I}_y$, $\frac{\partial \mathcal{I}}{\partial t} = \mathcal{I}_t$ it is possible rewrite Equation B.2 as:

$$\mathcal{I}_m = \mathcal{I}_x u + \mathcal{I}_y v + \mathcal{I}_t \tag{B.3}$$

Normally, the spatial and temporal derivatives are estimated by a difference approximation in a

cube of pixel intensity values [15, 18]. A 3×3 approximation is given below:

$$\mathcal{I}_x = \begin{bmatrix} \mathcal{I}(x+1,y,t) & - & \mathcal{I}(x,y,t) & + \\ \mathcal{I}(x+1,y,t+1) & - & \mathcal{I}(x,y,t+1) & + \\ \mathcal{I}(x+1,y+1,t) & - & \mathcal{I}(x,y+1,t) & + \\ \mathcal{I}(x+1,y+1,t+1) & - & \mathcal{I}(x,y+1,t+1) & \end{bmatrix} /4 \tag{B.4}$$

$$\mathcal{I}_y = \begin{bmatrix} \mathcal{I}(x,y+1,t) & - & \mathcal{I}(x,y,t) & + \\ \mathcal{I}(x,y+1,t+1) & - & \mathcal{I}(x,y,t+1) & + \\ \mathcal{I}(x+1,y+1,t) & - & \mathcal{I}(x+1,y,t) & + \\ \mathcal{I}(x+1,y+1,t+1) & - & \mathcal{I}(x+1,y,t+1) & \end{bmatrix} /4 \tag{B.5}$$

$$\mathcal{I}_t = \begin{bmatrix} \mathcal{I}(x,y,t+1) & - & \mathcal{I}(x,y,t) & + \\ \mathcal{I}(x,y+1,t+1) & - & \mathcal{I}(x,y+1,t) & + \\ \mathcal{I}(x+1,y,t+1) & - & \mathcal{I}(x+1,y,t) & + \\ \mathcal{I}(x+1,y+1,t+1) & - & \mathcal{I}(x+1,y+1,t) & \end{bmatrix} /4 \tag{B.6}$$

Equation B.3 is the equation of a line in velocity space and the motion estimate is constrained to lie on this line. To constrain the estimate to a single point it is necessary to employ further regularization constraints. Horn and Schunck proposed the smoothness constraint $\mathcal{I}_s$, which requires the motion vectors to vary smoothly over the neighbouring pixels:

$$\mathcal{I}_s^2 = \left(\frac{\partial u}{\partial x}\right)^2 + \left(\frac{\partial u}{\partial y}\right)^2 + \left(\frac{\partial v}{\partial x}\right)^2 + \left(\frac{\partial v}{\partial y}\right)^2 \tag{B.7}$$

A weighted combination of the smoothness and motion gradient constraints gives an error E, which must be minimized over the pixel locations (i, j) in the image:

$$E = \sum_i \sum_j \left(\mathcal{I}_m^2 + \lambda \mathcal{I}_s^2\right) \tag{B.8}$$

where λ is the weighting term. To minimize this equation for a motion vector at pixel location

(k, l), derivatives are taken with respect to u_{kl} and v_{kl} and the result is set to zero:

$$\frac{\partial E}{\partial u_{kl}} = 2(\mathcal{I}_{x_{kl}} u_{kl} \mathcal{I}_{y_{kl}} v_{kl} \mathcal{I}_{t_{kl}}) \mathcal{I}_{x_{kl}} + 2\lambda \left(\frac{\partial^2 u_{kl}}{\partial x^2} + \frac{\partial^2 u_{kl}}{\partial y^2} \right) = 0 \tag{B.9}$$

$$\frac{\partial E}{\partial v_{kl}} = 2(\mathcal{I}_{x_{kl}} u_{kl} \mathcal{I}_{y_{kl}} v_{kl} \mathcal{I}_{t_{kl}}) \mathcal{I}_{y_{kl}} + 2\lambda \left(\frac{\partial^2 v_{kl}}{\partial x^2} + \frac{\partial^2 v_{kl}}{\partial y^2} \right) = 0 \tag{B.10}$$

It can be seen that the second term on the RHS is the Laplacian of the pixel intensities, which Horn and Schunck approximate by:

$$\frac{\partial^2 u_{kl}}{\partial x^2} + \frac{\partial^2 u_{kl}}{\partial y^2} = u_{kl} - \bar{u}_{kl} \tag{B.11}$$

where $\bar{u}_{kl}$ is defined as:

$$\begin{aligned} \bar{u}_{kl} = \frac{1}{12} \; & [\; \mathcal{I}(i-1, j-1, t) + \mathcal{I}(i+1, j-1, t) + \\ & \mathcal{I}(i-1, j+1, t) + \mathcal{I}(i+1, j+1, t)] + \\ \frac{1}{6} \; & [\; \mathcal{I}(i, j-1, t) + \mathcal{I}(i, j+1, t) + \\ & \mathcal{I}(i-1, j, t) + \mathcal{I}(i+1, j, t)] \end{aligned} \tag{B.12}$$

and so from Equation B.9 and Equation B.10:

$$\frac{\partial E}{\partial u_{kl}} \approx 2 \left(\mathcal{I}_{x_{kl}} u_{kl} + \mathcal{I}_{y_{kl}} v_{kl} + \mathcal{I}_{t_{kl}} \right) \mathcal{I}_{x_{kl}} + 2\lambda (u_{kl} - \bar{u}_{kl}) \tag{B.13}$$

$$\frac{\partial E}{\partial v_{kl}} \approx 2 \left(\mathcal{I}_{x_{kl}} u_{kl} + \mathcal{I}_{y_{kl}} v_{kl} + \mathcal{I}_{t_{kl}} \right) \mathcal{I}_{y_{kl}} + 2\lambda (v_{kl} - \bar{v}_{kl}) \tag{B.14}$$

Where Equation B.8 is at a minimum, the derivatives must be 0, so:

$$(\mathcal{I}_{x_{kl}}^2 + \lambda)u_{kl} + \mathcal{I}_{x_{kl}}\mathcal{I}_{y_{kl}}v_{kl} = \lambda\bar{u}_{kl} - \mathcal{I}_{x_{kl}}\mathcal{I}_{t_{kl}} \quad \text{(B.15)}$$

$$\mathcal{I}_{x_{kl}}\mathcal{I}_{y_{kl}}u_{kl} + (\mathcal{I}_{y_{kl}}^2 + \lambda)v_{kl} = \lambda\bar{v}_{kl} - \mathcal{I}_{y_{kl}}\mathcal{I}_{t_{kl}} \quad \text{(B.16)}$$

This can be rewritten in matrix form as follows:

$$\begin{bmatrix} \lambda\mathcal{I}_{x_{kl}}^2 + \lambda & \mathcal{I}_{x_{kl}}\mathcal{I}_{y_{kl}} \\ \mathcal{I}_{x_{kl}}\mathcal{I}_{y_{kl}} & \mathcal{I}_{y_{kl}}^2 + \lambda \end{bmatrix} \begin{bmatrix} u_{kl} \\ v_{kl} \end{bmatrix} = \begin{bmatrix} \lambda\bar{u}_{kl} - \mathcal{I}_{x_{kl}}\mathcal{I}_{t_{kl}} \\ \lambda\bar{v}_{kl} - \mathcal{I}_{y_{kl}}\mathcal{I}_{t_{kl}} \end{bmatrix} \quad \text{(B.17)}$$

In order to get this in terms of u_{kl} and v_{kl}, the inverse of the 2×2 matrix on the LHS must be found. The determinant of the matrix is:

$$\lambda^2 + \lambda\mathcal{I}_{x_{kl}}^2 + \lambda\mathcal{I}_{y_{kl}}^2 \quad \text{(B.18)}$$

and so (B.17) can be written:

$$(\lambda^2 + \lambda\mathcal{I}_{x_{kl}}^2 + \lambda\mathcal{I}_{y_{kl}}^2) \begin{bmatrix} u_{kl} \\ v_{kl} \end{bmatrix} = \begin{bmatrix} \mathcal{I}_{y_{kl}}^2 + \lambda & -\lambda\mathcal{I}_{x_{kl}}\mathcal{I}_{y_{kl}} \\ -\lambda\mathcal{I}_{x_{kl}}\mathcal{I}_{y_{kl}} & \mathcal{I}_{x_{kl}}^2 + \lambda \end{bmatrix} \begin{bmatrix} \lambda\bar{u}_{kl} - \mathcal{I}_{x_{kl}}\mathcal{I}_{t_{kl}} \\ \lambda\bar{v}_{kl} - \mathcal{I}_{y_{kl}}\mathcal{I}_{t_{kl}} \end{bmatrix} \quad \text{(B.19)}$$

This gives 2 equations, one for u and one for v:

$$u_{kl} = \frac{(\mathcal{I}_{y_{kl}}^2 + \lambda)\bar{u}_{kl} - \mathcal{I}_{x_{kl}}\mathcal{I}_{y_{kl}}\bar{v}_{kl} - \mathcal{I}_{x_{kl}}\mathcal{I}_{t_{kl}}}{\lambda + \mathcal{I}_{x_{kl}}^2 + \mathcal{I}_{y_{kl}}^2} \quad \text{(B.20)}$$

$$v_{kl} = \frac{(\mathcal{I}_{x_{kl}}^2 + \lambda)\bar{v}_{kl} - \mathcal{I}_{x_{kl}}\mathcal{I}_{y_{kl}}\bar{u}_{kl} - \mathcal{I}_{y_{kl}}\mathcal{I}_{t_{kl}}}{\lambda + \mathcal{I}_{x_{kl}}^2 + \mathcal{I}_{y_{kl}}^2} \quad \text{(B.21)}$$

The Gauss-Seidel method can then be used to give an iterative solution:

$$u_{kl}^{n+1} = \frac{(\mathcal{I}_{y_{kl}}^2 + \lambda)\bar{u}_{kl}^n - \mathcal{I}_{x_{kl}}\mathcal{I}_{y_{kl}}\bar{v}_{kl}^n - \mathcal{I}_{x_{kl}}\mathcal{I}_{t_{kl}}}{\lambda + \mathcal{I}_{x_{kl}}^2 + \mathcal{I}_{y_{kl}}^2} \tag{B.22}$$

$$v_{kl}^{n+1} = \frac{(\mathcal{I}_{x_{kl}}^2 + \lambda)\bar{v}_{kl}^n - \mathcal{I}_{x_{kl}}\mathcal{I}_{y_{kl}}\bar{u}_{kl}^n - \mathcal{I}_{y_{kl}}\mathcal{I}_{t_{kl}}}{\lambda + \mathcal{I}_{x_{kl}}^2 + \mathcal{I}_{y_{kl}}^2} \tag{B.23}$$

Noting that:

$$\frac{(\mathcal{I}_{y_{kl}}^2 + \lambda)\bar{u}_{kl}}{\lambda + \mathcal{I}_{x_{kl}}^2 + \mathcal{I}_{y_{kl}}^2} = \bar{u}_{kl} - \frac{\mathcal{I}_{x_{kl}}^2\bar{u}_{kl}}{\lambda + \mathcal{I}_{x_{kl}}^2 + \mathcal{I}_{y_{kl}}^2} \tag{B.24}$$

and similarly for the vertical component, equations (B.22) and (B.23) are usually rearranged as follows [12, 15]:

$$u_{kl}^{n+1} = \bar{u}_{kl}^n - \frac{\mathcal{I}_{x_{kl}}(\mathcal{I}_x\bar{u}_{kl}^n + \mathcal{I}_{y_{kl}}\bar{v}_{kl}^n + \mathcal{I}_{t_{kl}})}{\lambda + \mathcal{I}_{x_{kl}}^2 + \mathcal{I}_{y_{kl}}^2} \tag{B.25}$$

$$v_{kl}^{n+1} = \bar{v}_{kl}^n - \frac{\mathcal{I}_{y_{kl}}(\mathcal{I}_x\bar{u}_{kl}^n + \mathcal{I}_{y_{kl}}\bar{v}_{kl}^n + \mathcal{I}_{t_{kl}})}{\lambda + \mathcal{I}_{x_{kl}}^2 + \mathcal{I}_{y_{kl}}^2} \tag{B.26}$$

B.2 Derivation of the Kanade-Lucas-Tomasi feature tracker

A common feature correspondence technique for motion estimation is the Kanade-Lucas-Tomasi (KLT) algorithm [87] that tracks region features. A detailed derivation of the basic equations is presented below based on that in [87]. Starting with the same assumption as Horn and Schunck [15], that intensity of an imaged point remains constant over space and time (Equation B.1) they state:

$$\mathcal{I}(\mathbf{x} - \mathbf{d}, t) = \mathcal{I}(\mathbf{x}, t + \Delta t) \tag{B.27}$$

They then state that there are two problems when tracking rigid regions using this assumption: reference region changes and non-rigid motion. The first problem, where the region being

tracked changes, is tackled through use of a dissimilarity measure. In their case they monitor the SSD residuals but other dissimilarity measures would also be appropriate [33, 113]. The second problem of non-rigid motion can be solved by tracking using a more complex motion model, such as the affine transformation. However, to detect more complex motion a larger region is required and their is a danger of over-parameterizing the system. They chose to track only displacement (u, v) for individual windows. The image model can be now defined as:

$$\mathcal{I}(\mathbf{x}, t + \Delta t) = \mathcal{I}(\mathbf{x} - \mathbf{d}, t) + n(\mathbf{x}) \tag{B.28}$$

where $n(\mathbf{x})$ is the image noise. Using an SSD function, the displacement vector $\mathbf{d}$ is chosen to minimize the residue error defined by the following double integral over the region window $\mathcal{W}$:

$$\epsilon = \int_{\mathcal{W}} [\mathcal{I}(\mathbf{x} - \mathbf{d}, t) - \mathcal{I}(\mathbf{x}, t + \Delta t)]^2 \, d\mathbf{x} \tag{B.29}$$

Tomasi and Kanade [87] now state that the linearization method present in [161] is the most efficiency way to minimize this residue assuming that *the displacement d is much smaller than the window size.* This is an important assumption and is not always discussed with regard to this tracker. The original derivation [161] suggested a weighting function over the region in Equation B.29, such as a Gaussian function to emphasize the centre of the region. However, this less commonly applied and has been removed for clarity.

The linearization method starts by assuming the the displacement vector is sufficiently small. The intensity function can now be approximated by its Taylor series truncated to the linear term.:

$$\mathcal{I}(\mathbf{x} - \mathbf{d}, t) = \mathcal{I}(\mathbf{x}, t) - \mathbf{g} \cdot \mathbf{d} \tag{B.30}$$

We can rewrite the residual (Equation B.29) as, with $h = \mathcal{I}(\mathbf{x}, t) - \mathcal{I}(\mathbf{x}, t + \Delta t)$:

$$\epsilon = \int_{\mathcal{W}} [\mathcal{I}(\mathbf{x}, t) - \mathcal{I}(\mathbf{x}, t + \Delta t) - \mathbf{g} \cdot \mathbf{d}]^2 \, d\mathbf{x} = \int_{\mathcal{W}} [h - \mathbf{g} \cdot \mathbf{d}]^2 \, d\mathbf{x} \tag{B.31}$$

This residue is now a quadratic function of the displacement $\mathbf{d}$ and, as a consequence, can be performed in close form. Differentiating the residual (Equation B.31) with respect to $\mathbf{d}$ and setting the result to zero yields the following equation:

$$\int_{\mathcal{W}} (h - \mathbf{g} \cdot \mathbf{d}) \, \mathbf{g} \, dA \tag{B.32}$$

Since $(\mathbf{g} \cdot \mathbf{d})\mathbf{g} = (\mathbf{g}\mathbf{g}^T)\mathbf{d}$ and $\mathbf{d}$ is assumed to be constant within the region $\mathcal{W}$, we have:

$$\left(\int_{\mathcal{W}} \mathbf{g}\mathbf{g}^T \, dA \right) \mathbf{d} = \int_{\mathcal{W}} h\mathbf{g} \, dA \tag{B.33}$$

This is a system of two scalar equations into two unknowns. It can be rewritten as:

$$G\mathbf{d} = e \tag{B.34}$$

where

$$G = \int_{\mathcal{W}} \mathbf{g}\mathbf{g}^T \, dA = \begin{bmatrix} g_x^2 & g_x g_y \\ g_x g_y & g_y^2 \end{bmatrix}$$

$$e = \int_{\mathcal{W}} h\mathbf{g} \, dA = \sum_{W} [\mathcal{I}(\mathbf{x} - \mathbf{d}, t) - \mathcal{I}(\mathbf{x}, t + \Delta t)] \begin{bmatrix} g_x \\ g_y \end{bmatrix}$$

Equation B.34 is the system to be solved for the tracking procedure. For every pair of adjacent frames, the matrix G can be computed from one frame by estimating the intensity gradients

and computing their second order moments. The vector e can be computed from the difference between the two frames, along with the gradient computed above. The displacement $\mathbf{d}$ is the vector to be solved with the system. It can be seen that the system (Equation B.34) has more unknowns that equations and this is the *aperture problem* discussed by Horn and Schunck [15]. However, the solution in this case comes from the rigid-body motion assumed by use of a region window $\mathcal{W}$. Considering the window $\mathcal{W}$, it is probable that different patches may allow us to compute different components of the displacement vector. By minimizing the residue across the entire window $\mathcal{W}$, we combine these measurements to produce a single displacement $\mathbf{d}$. This system (Equation B.34) is normally solved using a Newton-Raphson iteration method, which in practice achieves good accuracy within a small number of iterations.

The assumption that different patches allow computation of different displacement components is reasonable in highly textured areas but may fail under smaller amounts of texture. Therefore, a feature selection technique based on the system being solved (Equation B.34) is also presented in [87]. The advantage to the proposed feature selection is that it is not based on pre-conceived ideas about a good feature, such as corners or edges, but is based on the mathematical suitability of a region when solving the tracking system.

B.3 Derivation of a Linear Least-Squares Minimization

This section demonstrates the method used for a linear least-squares fit for a data set on a specific motion model. In this case the affine motion model is chosen. The affine motion transform is defined as follows:

$$\mathbf{x}' = A\mathbf{x} + \mathbf{d} \tag{B.35}$$

where

$$
\begin{aligned}
A &= \begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix} \\
\mathbf{d} &= (u, v)^T \\
\mathbf{x} &= (x, y)^T
\end{aligned}
$$

The error function to be minimized is chosen as the geometric distance between the correct displacement and the predicted displacement. The squared error function is minimized as follows:

$$\left[\left(E_x^2 + E_y^2\right)^{\frac{1}{2}}\right]^2 = E_x^2 + E_y^2 \tag{B.36}$$

Therefore, the sum of square errors to be minimized are:

$$q_x = \sum_{j=1}^{n} \left(x'_j - a_{11}x_j - a_{12}y_j - u\right)^2 \tag{B.37}$$

$$q_y = \sum_{j=1}^{n} \left(y'_j - a_{21}x_j - a_{22}y_j - v\right)^2 \tag{B.38}$$

where the two error functions are defined as:

$$E_x = (x' - a_{11}x - a_{12}y - u) \tag{B.39}$$

$$E_y = (y' - a_{21}x - a_{22}y - v) \tag{B.40}$$

To find the minimum a necessary condition is the q_x and q_y are minimum. Differentiating q_x by the three variables (a_{11}, a_{12} and u) within the equation and differentiating q_y by the three variables (a_{21}, a_{22} and v) within the equation gives:

$$
\begin{aligned}
\frac{\partial q_x}{\partial a_{11}} &= -2\sum_{j=1}^{n} X_j X'_j + 2a_{11}\sum_{j=1}^{n} X_j^2 + 2a_{12}\sum_{j=1}^{n} X_j Y_j + 2u\sum_{j=1}^{n} X_j \\
\frac{\partial q_x}{\partial a_{12}} &= -2\sum_{j=1}^{n} Y_j X'_j + 2a_{11}\sum_{j=1}^{n} Y_j X_j + 2a_{12}\sum_{j=1}^{n} Y_j^2 + 2u\sum_{j=1}^{n} Y_j \\
\frac{\partial q_x}{\partial u} &= -\sum_{j=1}^{n} X'_j + a_{11}\sum_{j=1}^{n} X_j + a_{12}\sum_{j=1}^{n} Y_j + nu \\
\frac{\partial q_y}{\partial a_{21}} &= -2\sum_{j=1}^{n} X_j Y'_j + 2a_{21}\sum_{j=1}^{n} X_j^2 + 2a_{22}\sum_{j=1}^{n} Y_j X_j + 2v\sum_{j=1}^{n} X_j \\
\frac{\partial q_y}{\partial a_{22}} &= -2\sum_{j=1}^{n} Y_j Y'_j + 2a_{21}\sum_{j=1}^{n} X_j Y_j + 2a_{22}\sum_{j=1}^{n} Y_j^2 + 2v\sum_{j=1}^{n} Y_j \\
\frac{\partial q_y}{\partial v} &= -\sum_{j=1}^{n} Y'_j + a_{21}\sum_{j=1}^{n} X_j + a_{22}\sum_{j=1}^{n} Y_j + nv
\end{aligned}
$$

Each equation has been written as three sums, grouping the variables to be solved into one equation. By setting each equation to zero and moving the first term from the RHS to the LHS we produce two systems of 3 equations, called the normal equations, with 3 unknowns. These can then be arranged into matrix form, as shown:

$$
\left(\begin{array}{ccc|c}
a_{11} & a_{12} & u & \\
\sum X_j^2 & \sum X_j Y_j & \sum X_j & \sum X_j X'_j \\
\sum Y_j X_j & \sum Y_j^2 & \sum Y_j & \sum Y_j X'_j \\
\sum X_j & \sum Y_j & n & \sum X'_j
\end{array}\right) \tag{B.41}
$$

$$
\left(\begin{array}{ccc|c}
a_{21} & a22 & v & \\
\sum X_j^2 & \sum Y_j X_j & \sum X_j & \sum X_j Y'_j \\
\sum X_j Y_j & \sum Y_j^2 & \sum Y_j & \sum Y_j Y'_j \\
\sum X_j & \sum Y_j & n & \sum Y'_j
\end{array}\right) \tag{B.42}
$$

Having 6 equations and 6 unknowns in two matrix systems it is now possible to used Gaussian elimination to calculate the linear least-squares fit for the given data to the affine motion model.

A linear least-squares fit may be calculated for other motion models in a similar manner. It may be necessary to use partial or full pivoting in the matrix system to achieve a solution. Using the rigid-body motion this was almost always necessary due to the construction of the matrix system.

Appendix C
Dervish landmine-clearance vehicle Test Data Sets

The data-sets used throughout this thesis were introduced specifically for testing of motion estimation for the Dervish landmine-clearance vehicle, apart from those separately discussed in Chapter 4. The Dervish data-sets are introduced in more detail here. All the test data sets have been included on the attached CD-ROM in MPEG movie format. Short examples of the original PPM images are also given. The complete PPM image sequences are available, with their ground truth, on the attached DVD-ROM.

C.1 Dervish Field-Trials

A number of separate Dervish field-trials took place during the course of the research for this thesis and it was possible to mount a downwards looking camera on the Dervish to capture video test footage. However, the field-trials were for the development of the mechanics and DECCA navigation system and it was not possible to control the motion of the Dervish in each field-trial. Over the course of the field-trials a considerable number of test runs contain motion where an error has occurred in one of the Dervish control systems. In practice this form of error would terminate any terrain sweep that Dervish was performing and the sweep would be restarted. Therefore, it is not necessary to review all the video sequences for suitability.

A Sony DCR-TRY890E Digital Video Camera was used to capture the field-trial video footage at 25fps. The high level of vibration within Dervish caused the automatic features on the camera to become confused, in particular almost continual and unnecessary refocusing was occurring. Furthermore, the final camera system will be operating with fixed-parameters and will not have the benefit of automatic camera features. Therefore, the digital video camera was operated with all parameters fixed. Due to physical constraints within the Dervish, the camera was mounted parallel to the ground and pointed at a mirror angled at 45 degrees. This restricts the possible ground-view slightly and dirt on the mirror can sometimes be seen within the image sequences.

This camera setup is originally detailed in Section 2.3.2. The video footage is stored in Sony MiniDV proprietary format and then converted to colour TIFF using the commercial image processing package Adobe Premier. These images were then converted to grey-scale PGM using the commercial image processing package JASC PaintShopPro.

After careful review of all the video footage, seven separate test data sets were selected. These are designed to provide a representative sample of all the conditions in which Dervish may operate. The sequences contain movement across concrete, tarmac, grass, mud and leaves are all contained within the data set. The motion in all the test data sets is close to the true motion expected under real operating conditions. The normal sweeping motion, where the rate of rotation is high compared to translation is contained within all data sets.

Sequence	Frames	Ground Scene	Motion
Concrete	450	concrete paving	slow sweep
LeavesStable	840	leaves over soil	sweep
LeavesUnstable	730	leaves over soil	sweep, rough ground
GrassRuler	500	grass / mud field with ruler	sweep
Grass	1000	grass / mud field	sweep
FastMove	300	grass / mud field	translation
Tarmac	1000	tarmac	sweep, vibration

Table C.1: *Selected sequences from the Dervish field-trial video sequences. This attempts to cover all motion seen within the field-trials.*

Manual Tracking

To obtain a form of ground-truth for the Dervish field trail data sets it was necessary to manually track the sequences. Initially a number of alternate motion estimation techniques were considered but it was not possible to verify that the techniques, matter how complex, were accurate. The manual track was performed using a simple Java Applet. It was decided to track three separate points through the sequence. While in theory it is desirable to track as many points as possible this is a tricky, time consuming task and there are a limited number of easily identifiable points within the image sequences. In practice, it was always possible to identify three good points to track over a limited number of frames. The centre of rotation was also manually identified for each sequence. The sequence of data for each point must then be transformed into movement and rotation about the centre of rotation. Equation C.1 shows the

mathematics of the transformation, where $\mathbf{R}$ is the matrix for rotation, $\mathbf{T}_c$ is the translation to the centre of rotation and $\mathbf{x}$ is the position vector. A StarOffice 5.2 Spreadsheet was set up to perform this transformation and this is included on the attached CD-ROM.

$$\mathbf{x}' = \mathbf{R}\left(\mathbf{x} - \mathbf{T}_c\right) + \mathbf{T}_c \tag{C.1}$$

The major problem presented by manual tracking was when significant image blur occurred. This was normally caused by vibration and made it very difficult to locate the target object. Therefore, the manual track is as accurate as the visual quality would allow for each individual frame and the accuracy of the motion estimates varies somewhat. When no vibration is present the accuracy can be considered ± 1-pixel. In the presence of significant vibration the accuracy drops to closer to ± 3-pixels. Another problem encountered was replacing the manually tracked features as they went out of view. It was very difficult to find easily identifiable target points on most of the sequences. Over an extended period the overall accuracy of the manual tracking could not be guaranteed and shortened versions of the sequences were tracked. Table C.2 shows the information on the manually tracked within sequences.

Sequence	**Tracked Frames**	**Centre of Rotation**	**Accuracy**
Concrete	200	(203, 120)	high
LeavesStable	300	(215, 140)	high
LeavesUnstable	200	(175, 130)	high
GrassRuler	240	(170, 155)	medium
Grass	179	(178, 130)	medium

Table C.2: *The sequences that were manually tracked are shown with the number of frames tracked and the centre of rotation used for processing.*

The first frame from each of the test sequences is shown in Figure C.1 giving an indication of the image content for that sequence.

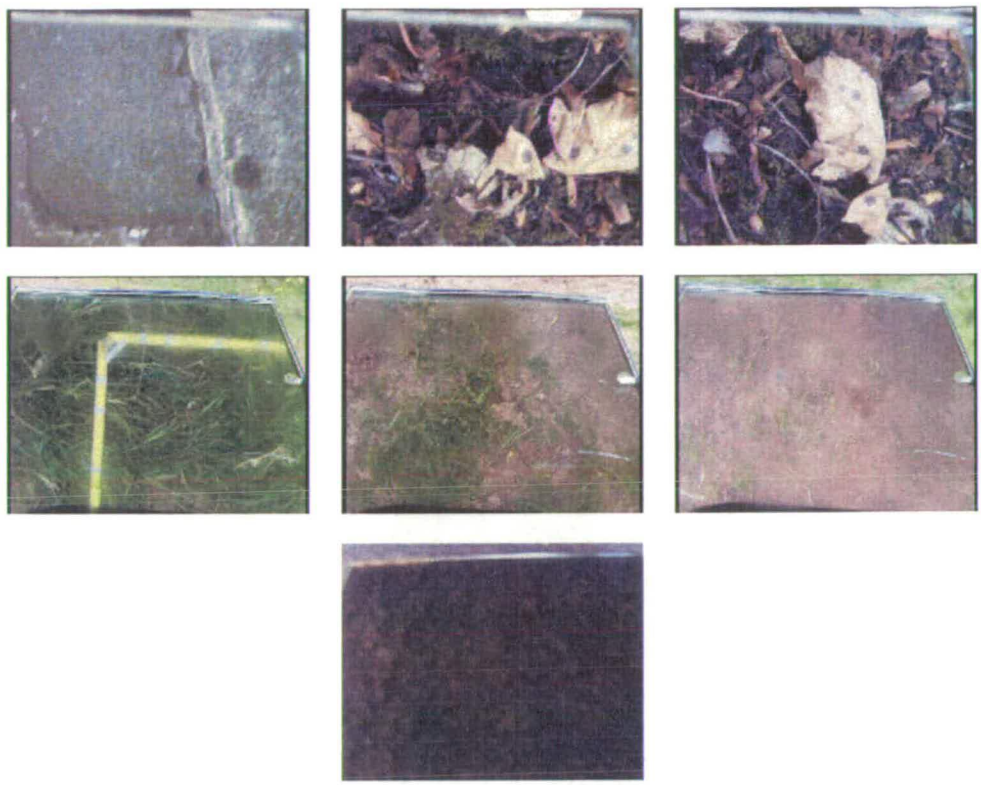


Figure C.1: *Figure showing the first frame from each of the sequences from the Dervish field-trials. Going from left to right, top to bottom the sequences are Concrete, LeavesStable, LeavesUnstable, GrassRuler, Grass, FastMove and Tarmac. All sequences apart from FastMove and Tarmac have been manually tracked.*

C.2 Dervish Simulator

Overview

The problem of obtaining ground-truth for the Dervish field-trial video sequences combined with the lack of any new field-trials since January 2001 presented a problem testing the accuracy of the system over an extended period. Since the accuracy over an extended period is a fundamental requirement a series of simulated image sequences were generated. The aim for the Dervish simulator was to reproduce the output seen within the Dervish field-trial video sequences. No attempt was made to model the underlying systems due to the complexity of the system. In particular the mechanics of the Dervish and the DECCA-based navigation system are far too complex to allow a simple model to be reproduced. It may be a pertinent exercise to produce models for each of the Dervish systems to allow more accurate analysis of all the navigation system but that is outside the scope of the research presented in this thesis.

C.2.1 Motion parameters, Noise Sources and Errors

The motion parameters have all been determined from analysis of the Dervish field trial video sequences. It is necessary that the same motion parameters be used for the Dervish simulator. The first stage for the Dervish simulator was to identify separate error and noise sources within the system. All noise and error sources are modelled as zero-mean signal-independent Gaussian distributed. It may be possible, through more complex analysis, to determine more suitable noise distributions but this was considered unnecessary for this first-order model. The following simulation parameters were identified:

1. Error between Dervish position and DECCA-net
2. Error between Dervish requested motion and actual motion (*circular wobble*)
3. Image capture noise
4. Image blur due to Dervish vibration
5. Image illumination variation

The first two forms of error are introduced to allow realistic movement of the simulated image sequence, including the *circular wobble* seen within the test data sets. This now determines the camera movement across the artificial terrain. Modelling of camera optics was considered but it was decided that the video sequences showed mainly planar motion with little or no depth variation. Therefore, no projection of the terrain takes places and the motion is directly applied to the image plane. Given that the terrain is in fact a 2D picture this seems an appropriate simplification to make. Therefore, the main source of noise within the image is modelled as image capture noise. The image blur due to Dervish vibration remains a significant problem but it is not clear at this stage if it is going to remain an issue. Extensive work was undertaken to remove much of the vibration and it appears now to only be a problem on hard surfaces, such as concrete and tarmac. However, it is not necessary operate with high accuracy while traversing tarmac or concrete as no landmine-clearance operations are likely to be taking place. However, the model attempts to reproduce the issue to allow some testing to take place. Finally, illumination changes, including direction and intensity, are allowed. Not having 3D information on the terrain forces these to be heavily simplified.

C.2.2 Movement

The Dervish movement in the simulator is represented by a DECCA Net object with Dervish object following this. The DECCA Net object has a pre-programmed path which it follows; if Dervish strays too far from the node it pauses until Dervish regains position. The Net error is monitored by telemetry in the real system for this purpose and provides a good indication of how accurately Dervish is tracking the Net. The Dervish object attempts to follow the Net as accurately as possible but is restricted by the maximum translation and rotation possible by Dervish in the time frame; these have been determined from the Dervish field-trial data sequences. The Dervish object is also effected by the circular wobble seen in the field-trials and some general error within the requested motion, modelled as Gaussian noise.

The image processing is handled by the VS_frame class libraries from the Vision group in the Department of Electronics and Electrical Engineering. These libraries provide good functionality for all the tasks. The only drawback is that only bilinear interpolation is available and this will lead to increased accuracy in the feature tracking process should it also used bilinear interpolation. However, the aim of the simulator is not to provide accuracy estimates for the individual features and this should not be a significant problem.

There are a large number of model parameters included in the system to allow testing of a wide spread of techniques. The model parameters were varied in testing to cover many of the possible situations that the Dervish will face. Some test sequences are available on the attached CD-ROM to give an indication of the types of image sequences that have been tested. The simulator code has also been included on the attached CD-ROM and suitable parameters are indicated within the code.

Appendix D
Description of Hardware

This appendix provides a more detailed description of the hardware development platform used in the research, together with relevant references for extensive information sources. Section 5.4 provided a summary of the development platform structure and Section 5.5 provided detail on the following topics:

- Section 5.5.1 Host-DSP Transfer Environment
- Section 5.5.2 Data Transfer Format
- Section 5.5.3 Clock speed and internal timer
- Section 5.5.4 Zero-overhead loops
- Section 5.5.5 SHARC Memory Structure
- Section 5.5.6 In-place / Out-of-place Regions

D.1 Bittware Hammerhead development platform

The Bittware Hammerhead development platform is based around the Analog Devices SHARC DSP. It has four DSPs, external DRAM and communications, shown in Figure D.1. The communications have been simplified to show only the SHARC*fin* ASIC. The main form of external communication is a PCI bus for interfacing to a PC host. It also has two PMC+ slots that could be used to allow direct inclusion of a frame-grabber. A central bus operating at 33MHz is controlled by the SHARC*fin* ASIC and the majority of data transfer will take place through this. The SHARC processors are also connected in a ring through their Link ports, which provide high speed inter-processor communication. The Link ports are capable of providing more than 240Megabytes per second.

The most likely bottleneck within the Bittware Hammerhead system is the SHARC*fin* ASIC. This controls all external access, PCI or PMC+, through the main bus. The main bus is also

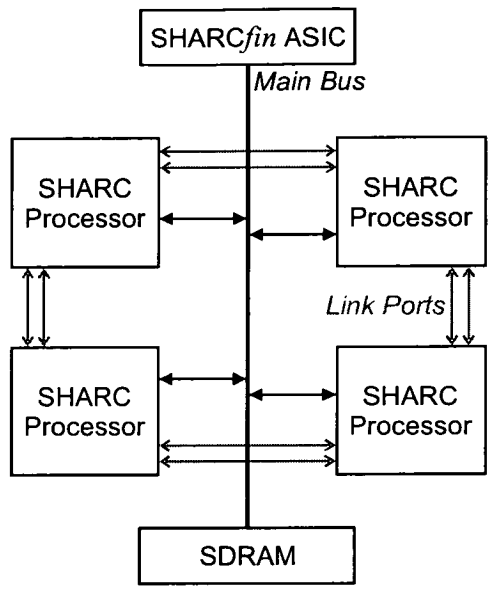


Figure D.1: *A simplified diagram of the structure of the Bittware Hammerhead quad-SHARC development board. The SHARCs and SDRAM are all connected through a central bus, which is controlled by the SHARCfin ASIC. The SHARCfin ASIC controls external access as well, such as the host PCI bus. The SHARC processors are also connected in a ring through their high-speed Link Ports, allowing direct processor-to-processor communication.*

used by all the processors to access the SDRAM and only one processor may have access at a time. Therefore, it is important to avoid SDRAM access while any external data transfer, such as image download, is occurring. If multiple access appears unavoidable it may be possible to make use of the inter-processor Link ports to avoid the repeated SDRAM access. However, the SHARC*fin* controlled bus operates at 66MHz with a bus-width of 32-bits. The packed data size per frame is 25KBytes and bus bandwidth should be capable of handling the low-resolution images at 5 frames per second without too much trouble.

Although this platform would be suitable for final implementation it is really designed as a development platform. It may reduce the cost further if a simpler PCB were employed with the same basic components.

Host-DSP Transfer Environment

As stated in Section 5.1 the best way of handling the Host-DSP transfer is to allow the Host to perform all the memory transfer options but to have the signalling variables on the DSP. This continues to allow the DSP control of the timing but removes from it any computational

cost associated with the download. The Host program has been coded in Visual C++ and can be found on the attached CD-ROM. To summarize, it downloads a program to each DSP and then starts them running. From then on it waits to download a frame when requested and then upload the results. The Bittware libraries contain a number of built-in functions for uploading and downloading any form of data. The uploads and downloads are all to DSP memory but it is possible to transfer from both the DM and PM memory banks. It is also possible to map the external SDRAM to a DSP memory address and transfer directly to the SDRAM.

The data transfer routines are not designed for final system use and operate too slowly. However, since it is possible to transfer data directly to the SDRAM, it is possible to transfer sections of an image sequence before starting the DSP program. The final data transfer routines will have to be written when the frame-grabber has been chosen. However, a data transfer rate of 125KBytes per second should be achievable.

Data Transfer Format

Section 5.2 introduced the packing routines and having decided to operate using out-of-place regions it is possible to pack down to 8-bits rather than 16-bits because the extra unpacking step can be performed at the region extraction stage. A packing routine, to pack the 8-bit data into a 32-bit integer was written on the host side. A 16-bit packing routine is also present. The only difficulty with this routine is the number handling within C++. It is not possible to guarantee a set number of bits using the standard data types (char, short, int) so specific data types must be used that reduce portability.

D.2 Analog Devices ADSP-21160N SHARC

The Analog Devices SHARC DSP is designed as a simple architecture with an internal structure that is designed to perform basic operations very quickly. This is in contrast to modern general-purpose processors, such as the Motorola PowerPC or the Intel Pentium 4, where extensive use of pipelines, branch prediction, extremely high clock speeds and up to three levels of cache make optimization a complex business. The internal structure of the SHARC is shown in Figure D.2. Further, extensive details can be found in the SHARC user manual [162].

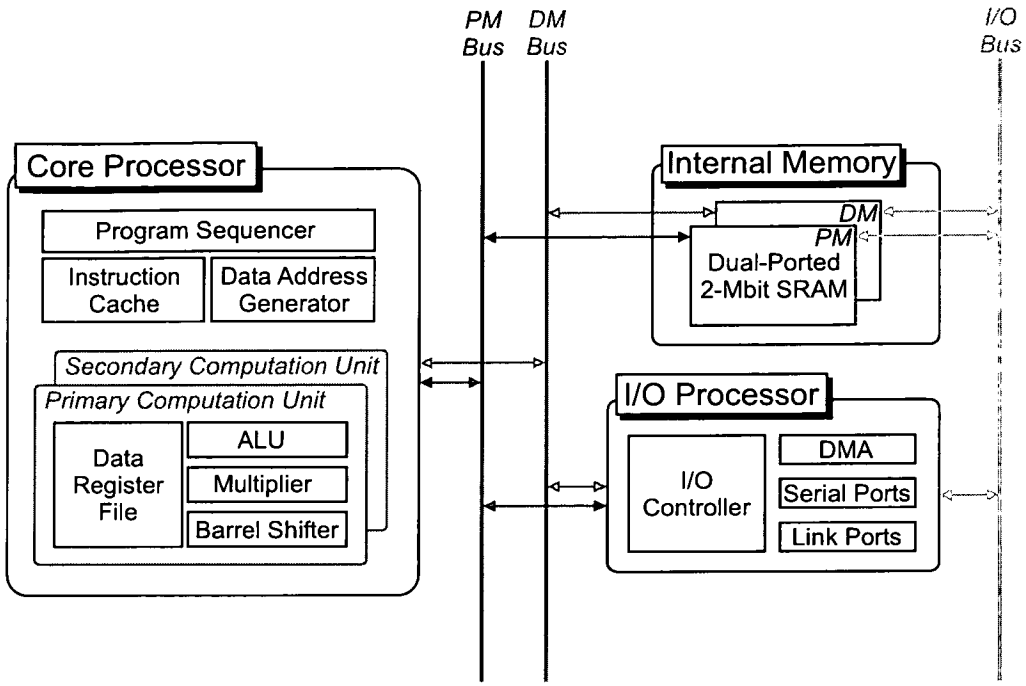


Figure D.2: *A simplified diagram for the structure of the ADSP21160 SHARC processor. The three data buses (PM, DM, I/O) can be seen, along with the dual-ported internal SRAM, which is divided into two 2Mbit blocks. The secondary computation unit is used when performing SIMD operations.*

Core Processor

The core processor has a program sequencer, two data address generators, data register file and instruction cache. Section 5.5.3 already discussed the internal timer and clock speed of the SHARC processors. The inclusion of two data address generators and program sequencer allow maximum efficiency in operation. For example, it is possible to fetch an instruction (cache) and two data operands (DM, PM). The data address generators also implement circular buffers, for delay lines in digital signal processing, and bit-reversal. There are four external hardware interrupts with the option for very low-overhead handling routines. To increase efficiency in the handling of hardware interrupts, the processor has an alternate register bank. Finally, the 32-word instruction cache is primarily used for the zero-overhead loops discussed Section 5.5.4. For this reason, the cache is selective and only stores instructions whose fetch conflict with a program memory data access. This allows excellent use of the, otherwise, small cache size and, with careful loop design, is sufficient for most applications.

Computation Units

The DSP has two computation units with a multiplier, ALU and barrel shifter in each. The multiplier, ALU and barrel shifter perform single-cycle operations on a number of data formats, including 32-bit floating-point. They are connected in parallel to the data register file and the output from any one can be the input for any other. A number of *multifunction* operations, where the multiplier and ALU operate simultaneously in one-cycle are possible. The two types of multifunction operations are: dual add/subtract and parallel multiply/ALU. However, to achieve multifunction operation it is necessary for the input to the ALU and multiplier to come from certain registers but the output may go to any register. For this reason, careful assignment of registers is necessary in multifunction operations. Although multifunction operations are very useful they require careful design to achieve optimum efficiency.

The two computation units allow SIMD operation of the multiplier/ALU/barrel shifter. The two data buses (DM, PM) are 64-bits wide allowing SIMD operation on 32-bit data values with the two computation units. The SIMD operations must be on 64-bit aligned data and the data must be in a continuous, sequential block. There are no further constraints and besides a MODE1 register set, operation is transparent to the user.

Internal Memory

The two dual-ported banks of internal memory allow very good access to the 4Mbits of SRAM. The dual-porting allows single-cycle simultaneous access by both the core processor and I/O processor, providing truly transparent background data transfer. The program memory (PM) and data memory (DM) banks are provisionally for program code (48-bits) and data (32/40/64-bits). However, the internal structure is defined by the user and data types can be mixed within the same memory bank. It is possible to mix data and code segments within the PM, allowing extra data space. Furthermore, once instructions have been cached the PM data can be loaded alongside the DM data for single-cycle dual-load operation. The memory format for the research in this project allowed 32KBytes of code in the PM. The remaining 32KBytes in the PM was used for data along with the 64KBytes of data memory in the DM, totalling 96KBytes of data memory. The memory description file, called the Linker Description File (LDF), used in this project is included on the attached CD-ROM.

I/O Processor

The I/O processor contains the serial ports, Link ports and DMA controller. On the Bittware Hammerhead development board, which contains bulk SDRAM, the DMA controller is the most useful as zero-overhead transparent data transfer can take place into internal SRAM at very high speed. On the Bittware Hammerhead development board the data transfer between SDRAM and internal SRAM is almost 44MBytes per second. A single transfer of a test image (360×288) from external memory to internal memory took 12700 clock cycles when tested on the Bittware Hammerhead platform.

D.3 Software Development

The software development went through a number of stages. The first stage used the Vision group libraries, which included implementations of block tracking, KLT, Horn and Schunck and Uras differential optical flow, to test out a number of ideas. Once the initial concepts had been tested a more optimized Unix implementation was developed that allowed extensive testing of the correlation-based feature tracker. The final stage was DSP implementation, including the development of the Host software under Windows 98. The attached CD-ROM also has much of the code used. This is split into three areas: Unix implementation, Host side software and SHARC software. The Unix implementation provides a fully working Unix version that can be compiled with a standard C++ compiler. The Host side software and SHARC software required special software drivers and the Bittware Hammerhead board to operate. The following sections describe each in more detail.

Unix Implementation

A fully working Unix implementation for the correlation-based feature tracking, feature management and passive navigation stages. There are two separate versions that were used in Chapter 6 and Chapter 7 respectively. The Chapter 6 version tracks a single block through any of the Dervish field trial sequences and reports accuracy of the track. The Chapter 7 version uses the feature management and passive navigation stages to track, with up to 48 features, through the synthetic sequences. It is possible to run the Dervish field trial sequences but due to the lower accuracy of the manual estimates there is little information to be gained.

Host / DSP Implementation

The Host implementation is given on the attached CD-ROM and has been discussed in previous sections. The DSP implementation is under VisualDSP++ v1.0. Selected files from the implementation are given on the attached CD-ROM. The implementation on the DSP was only undertaken once the specific algorithm had been designed and tested on the UNIX. This helps minimize debugging problems on the DSP. Key examples from the DSP implementation are given on the attached CD-ROM.

The main process for DSP implementation was to start with a basic DSP implementation in C and to develop a test set for the local routine. This could be used to confirm that the original C implementation and any future optimizations work correctly. Once the original C implementation had been tested the optimizing compiler was used and the assembler output examined. Only the loop structures were examined as the rest of the assembler, performing tasks such as stack maintenance, do not make up sufficient computation to be worth optimizing beyond that of the compiler.

In many cases after examining the assembler output from the compiler it was possible to determine a simple way of rearranging the the C code to allow the optimizing compiler to produce near-optimal output. In cases were this was not possible the next stage was to write inline-assembler using the ASM statement in VisualDSP++. This allows small sections of assembler to be written in the C file and inlined directly. Only in a small number of cases where a larger amount of rewriting was necessary was the complete function written in assembler.

Appendix E
Summary of System Recommendations

This appendix is to provide a summary of the conclusions reached during the research presented in this thesis. This is intended to provide a starting point for the implementation of a Dervish vision-based short-range navigation system.

Camera set-up

- Sony 3-CCD DCR-TRV890 Digital Video Camera
- Horizontal ground resolution of 1.7mm per pixel
- Vertical ground resolution of 1.4mm per pixel
- Video output in 8-bit grey-scale 360×288

Hardware Platform

- Bittware Hammerhead development board with PCI interface
- Four ADSP-21160N SHARCS
- 16MBytes SDRAM
- Dual PMC+ connectors for integration of peripherals (i.e. frame-grabber)

Correlation-based feature tracking

- Two-level hierarchical search pattern with overlapping L_1 search and rotation reference set (*Section 6.3*)
- 3×3 mean filter with a subsampling rate of 3 (*Section 6.34*)

- L_2 search of ± 8 pixels and [-0.06, 0.12] radian search @ 0.06 radians
- L_1 search of ± 4 pixels and [-0.04, 0.04] radian search @ 0.02 degrees
- Block size of 27×27 pixels (*Section 6.6*)
- Optimized Zero-mean Normalized Correlation Coefficient (ZNCC) (*Section 6.4*)
- Optimized Extraction of rotated regions with bilinear interpolation (*Section 6.5*)
- track-initiated DLS reference block update strategy with $\theta = 0.9$ (*Section 7.2.2*)

Feature Management and Passive Navigation

- Regular feature distribution up to a maximum of 48 features (*Section 7.2.1*)
- Feature initialisation based upon regular distribution (*Section 7.2.1*)
- Reference update based feature dissimilarity measures (*Section 7.2.3*)
- Robust two-stage minimization for passive navigation (*Section 7.3*)
- X84 based feature rejection for rotation outliers (*Section 7.3.3*)

Final System Design

- Operation at 5 frames per second
- Translation in millimetres (accuracy ± 0.8cm)
- Rotation in degrees (± 1 degree accuracy)
- Video stream and feature information output