Smart VLSI Micro-Sensors for Velocity Estimation inspired by Insect Vision

by

Xuan Thong Nguyen B.E. (Hon. I), B.Sc.

Thesis submitted for the degree of

Doctor of Philosophy

The University of Adelaide

Faculty of Engineering

The Department of Electrical and Electronic Engineering

March 1996

To Mai Trinh for life companionship

&

To my beloved parents, who taught me

the value of education

and

the beauty of hard work

Contents

Abstract

Motion detection is a challenging problem in machine vision. If motion is successfully detected, the obtained information can be potentially utilised in many real-time applications. However, existing techniques for motion detection, like optical flow computation, and optical motion detectors, such as CCD camera, are far from being useful in real-time applications due to some major difficulties such as the computational bottleneck. To improve the real-time applications of visual motion detectors a new approach called *smart sensing* has been pursued where VLSI smart micro-sensors have been designed for the specific task of motion detection.

In this thesis, the smart sensing paradigm is exploited in two aspects. Firstly, insect vision principles have been applied to the main mechanism for motion detection. Secondly, advanced VLSI technologies have been employed for designing *smart micro-sensors* in which the imager and processor are integrated into one monolithic device.

A neural model for motion detection, named *template model*, which consists of interpreting local changes in intensity in order to infer directional motion information, has been studied and analysed. The analysis consolidates the validity of the model with the support of electrophysiological evidence from the study of insect vision, and of a mathematical formulation. This formulation and a heuristic investigation are together employed to identify responses due to real coherent motion, called *directionally motion-sensitive templates.* These templates can be used as features for representing motion trajectories in the spatio-temporal domain. To exploit the benefits of using the feature representation of motion trajectories without costly computational and hardware implementation phases, novel tracking algorithms to estimate the relative velocities of moving stimuli have been developed.

In an attempt to transfer knowledge on insect vision into hardware, particularly the template model, advanced VLSI technologies have been used to build a first concept-demonstrating and then a second test-facilitating sensor. Results obtained by testing the sensors in real time confirm that the sensors function correctly as expected from the simulation including the responses of the directionally motion-sensitive templates and the relative errors achieved from velocity estimation. However, further implementation works are needed to integrate tested modules into one device and to make it available to a wide range of applications.

Statement of Originality

I hereby declare that this work contains no material which has been accepted for the award of any other degree or diploma in any university or other tertiary institution and that, to the best of my knowledge and belief, contains no material previously published or written by another person, except where due reference has been made in the text. I also give consent to this copy of my thesis, when deposited in the University Library, being available for loan and photocopying.

Xuan Thong Nguyen

25 March 1996

Acknowledgements

The author is indebted to a number of people without whom this work would not have been possible. Firstly, I would like to deeply thank my supervisors, Prof. R. E. Bogner, Dr. A. Bouzerdoum, and Prof. K. Eshraghian for their guidance, support, and encouragement. Prof. Bogner advised me in all aspects. Dr. Bouzerdoum, with whom I was closely collaborating, provided me opportunities to discover many fascinating aspects of the insect visual system and to discuss different neural models of motion detection, and suggested new ideas.

It is very fortunate for me to be involved in a team project (bugeye team) where we were enjoyably sharing responsibility and working together in order to achieve the goal of designing smart micro-sensors successfully. In particular, Mr. Ali Moini and Dr. Andre Yakovleff, who are two of early members of the team, suggested some useful ideas in designing the template and velocity estimation processors for the smart micro-sensor II in Chapter 8. In fact I cooperated with Andre, who is also one of readers of my numerous drafts, in proposing the VLSI architecture of the stair-step tracking algorithm, and in discussing the best way to use the smart micro-sensors. Thank you both for your collaboration. Thanks also to my other colleagues in the bugeye team including Mr. Dereck Abbott, Mr. Andrew Blanksby, Mr. Richard Beare, Mr. Andrew Beaumont-Smith, Mr. Ryan Fok, and Mr. Dino Papa. Mr. Dino Papa also assisted this work by simulating the template and velocity estimation processors in Chapter 8.

I would like also thank Mr. Ray Peterson from the Advisory Centre for University Education and Mr. Charlie Carmine in the department for their effort in correcting English in my early drafts, and Mr. Mathew Parry for his early works in the optics for the sensors.

The support of the Australian Research Council and of the University of Adelaide is gratefully acknowledged.

I would like also thank my parents, who are far away but always support and encourage me to pursue high education, my brother for bringing me to Australia so I have had the chance to feel free and to work towards the task.

Lastly, but not least, I would like to thank my wife, Mai Trinh for her understanding, endless patience and support, and our coming first baby who has recently made our life more colourful and joyful.

Publications

The following is a list of publications which are related to the research reported in this thesis.

[1] Nguyen, X.T., Eshraghian, K., Moini, A., Bouzerdoum, A., Yakovleff, A., Abbott, D., and Bogner, R.E. (1993), "An Implementation of A Smart Visual Micro-Sensor based upon Insect Vision," *Proc. 12th Australian Microelectronics*, 129-134, Gold Coast, Queensland, 5-8 October.

[2] Nguyen, X.T., Bouzerdoum, A., Bogner, R.E., Eshraghian, K., Abbott, D., and Moini, A. (1993), "The Stair-Step Tracking Algorithm for Velocity Estimation." *Proc. First Australian and New Zealand Conference on Intelligent Information Systems,* 412-416, Perth, Australia, 1-3 December.

[3] Nguyen, X.T., Yakovleff, A., Moini, A., Eshraghian, K., Bouzerdoum, A. and Bogner, R.E. (1994), "VLSI Architecture of a Low Computation Load Processor for a Visual System," *Proc. The European Design & Test Conference 94*, User Forum, 85-89, Paris, France, February-March.

[4] Nguyen, X.T., Bouzerdoum, A., Yakovleff, A., Moini, A., Bogner, R.E., and Eshraghian, K. (1994), "A VLSI Robotic Micro-Sensor: Range-Finder from Self Motion," *Proc. IEEE Int. Conf. Mechatronics and Machine Vision in Practice,* 78-83, Toowomba, QLD, Australia, 13-15 September.

[5] Nguyen, X.T., Bouzerdoum, A., Bogner, R.E., Moini, A., and Eshraghian, K. (1995), "Feature Representation of Motion Trajectories," *Proc. IEEE Int. Conf. Neural Networks*, **6**, 2922-2927, Perth, Western Australian, 27 November -- 1 December.

[6] Nguyen, X.T., Bouzerdoum, A., and Moini, A. (1996), "Velocity Measurement Using a Smart Micro-Sensor," (To appear in) *Int. Sym. Robotics Cybern.*, Lille, France, 9 -12 July.

[7] Yakovleff, A., Moini, A., Bouzerdoum, A., Nguyen, X.T., Bogner, R.E., and Eshraghian, K. (1993), "A Micro-Sensor based on Insect Vision," *Proc. Workshop on Computer Architecture for Machine Perception,* 137-146, USA,15--17 December.

[8] Yakovleff, A., Nguyen, X.T., Bouzerdoum, A., Moini, A., Bogner, R.E. and Eshraghian, K. (1994), "Dual-Purpose Interpretation of Sensory Information," *Proc. IEEE Int. Conf. Robot. Autom.,* 1635-1640, San Diego, 8-13 May.

[9] Yakovleff, A., Abbott, D., Nguyen, X.T., Eshraghian, K. (1995), "Obstacle Avoidance and Motion-Induced Navigation," *Proc. Workshop on Computer Architecture for Machine Perception (CAMP'95),* 384-393, Como, Italy, 18--20 September.

[10] Bouzerdoum, A., Bogner, R.E., Eshraghian, K., Moini, A., Yakovleff, A., and Nguyen, X.T. (1993). "A VLSI front-end visual processor based on insect vision," *Sensory Stratagems: From Living Eyes to Seeing Machines,* Canberra, 7--11 February.

[11] Bouzerdoum, A., Moini, A., Yakovleff, A., Nguyen, X.T., Bogner, R.E., and Eshraghian, K. (1994), "A Smart Visual Micro-Sensor," *Proc. IEEE Int. Conf. Systems, Man, and Cybernetics*, 276-279, San Antonio, Texas, 2-5 October.

[12] Abbott, D., Moini, A., Yakovleff, A., Nguyen, X.T., Blanksby, A., Kim, G., Bouzerdoum, A., Bogner, R.E., and Eshraghian, K. (1994), "A New VLSI Smart Visual Sensor for Collision Avoidance inspired by Insect Vision," *Proc. SPIE's Intelligent Vehicle Highway Systems Symposium*, **2344**, 105-115, Boston, MA, 2-4 November.

[13] Abbott, D., Yakovleff, A., Moini, A., Nguyen, X.T., Blanksby, A., Beare, R., Beaumont-Smith, A., Kim, G., Bouzerdoum, A., Bogner, R.E., and Eshraghian, K. (1995), "Biologically Inspired Obstacle Avoidance - A Technology Independent Paradigm," *Proc. SPIE's Mobile Robots X*, **2591**, 2-12, Philadelphia, Pennsylvania, 23-44 October.

[14] Moini, A., Bouzerdoum, A., Eshraghian, K., Yakovleff, A., and Nguyen, X.T. (1995), "The Architecture of an Insect Vision Based VLSI Motion Detection Chip," *Proc. 13th Australian Microelectronics*, 68-73, Adelaide, 16-19 July.

[15] Moini, A., Bouzerdoum, A., Eshraghian, K., Yakovleff, A., Nguyen, X. T., Blanksby, A., Beare, R., Bogner, R. E., and Abbott, D. (1995), "The Architecture and Analog Building Blocks of an Insect Vision Based VLSI Motion Detection Chip," Submitted to IEEE JSSC.

[16] Moini, A., Bouzerdoum, A., Eshraghian, K., Yakovleff, A., and Nguyen, X. T. (1995), "The VLSI Implementation of an Insect Vision Based Motion Detection Chip," Submitted to *OFOGH, the Journal of Computer Science and Engineering*.

List of Abbreviations

1D	One Dimensional	63
2D	Two Dimensional	1
3D	Three Dimensional	12
BJT	Bipolar Junction Transistor	142
BT	Backward Tracking	91
CCD	Charge-Coupled Device	2
CMOS	Complementary Metal Oxide Silicon	3
DMST	Directionally Motion-Sensitive Template	50
DSF	Directional Selectivity Function	48
DSMD	Directionally-Selective Motion-Detecting	32
EMD	Elementary Motion Detector	32
EPLD	Erasable Programmable Logic Design	139
FIFO	First-In First-Out	94
FT	Forward Tracking	93
GaAs	Gallium Arsenide	120
GRIN	Gradient Index	168
IC	Integrated Circuit	10
I/O	Input/Output	129
LMC	Large Monopolar Cell	29
MOS	Metal Oxide Silicon	3
PCT	Position Conjugate Template	67
RAM	Random Access Memory	121
SSD	Sum of Squared Difference	17
SST	Stair-Step Tracking	89
VLSI	Very Large Scale Integration	3
$\downarrow$	Decrease in Intensity	42
$\uparrow$	Increase in Intensity	42
$-$	No-Change in Intensity	42

List of Figures

List of Tables

Chapter 1

Introduction

Insects fly expertly among obstacles in a three-dimensional cluttered world, chase mates or prey, or land and take off to escape capture. The lesson for robot vision is obvious: considering how underdeveloped is our technology for visual processing, almost anything we can learn from insect vision is likely to be of use. Their performance shows us what to attempt but not how to do it; for that, they have many clever mechanisms for us to discover.

[Adrian Horridge, 1992]

1.1 Background and Motivation

1.1.1 Insect Vision

Unlike humans, insects have compound eyes which are two-dimensional (2D) arrays of tiny eyes, called *ommatidia*. Each tiny eye looks in a different direction and has its receptive field overlapping with those of neighbouring eyes, thus increasing the sensitivity to a diffuse source. Importantly, behind each of these eyes, there exists a detecting channel which may interact with neighbouring channels at different processing levels; each of the levels has a distinct neuronal layer. In other words, the insect visual system is architecturally structured into parallel processing layers and columns (Järvilehto 1985).

Insect vision is low-level vision; that is, insects mostly just *see* without further attempt to recognise *what they are seeing* (Beare et al. 1995). At this level of complexity, the insect visual system has not evolved far beyond the ability to detect directional motion, which can be performed by a number of neurons (Horridge 1987).

In fact, the sensitivity to directional motion is the characteristic of the visual systems of insects. Experimental evidence shows that most insects exhibit a tendency to turn in the direction of a drum's rotation when they are positioned in the centre of a rotating striped drum (Srinivasan 1992). Insects rely primarily on motion detection for navigating their course of flight, landing, and discriminating objects at different distances (Reichardt & Poggio 1976). It is amazing that without a complex brain like that of human beings, insects can

sufficiently detect directional motion, while powerful machines still fail to do the job in real-time (Horridge 1987; Franceschini 1992). This paradox has given rise to the hope of applying principles of insect vision to many real-time applications involving motion detection, such as mobile robot vision. In fact, a model which can be constructed based on insect vision suggests that computational load can be reduced by incorporating neuron-like parallel signal processing in the sensor itself, thereby alleviating subsequent processing requirements.

Motion processing in insects has been extensively studied over almost 40 years. Much of our knowledge of the mechanism of motion detection in insects has been accumulated from electrophysiological experiments on flying insects such as flies, bees, dragonflies, locusts, and others. As a result, a number of neural models of the mechanism of motion detection in insects have been proposed. Although some architectural differences do exists among these models (see, e.g., Bouzerdoum 1991; Beare et al. 1995; and Nguyen et al. 1995), they all agree that the mechanism of motion detection in insects is directionally sensitive. There exist some motion-sensitive neurons which produce a strong response when a visual stimulus moves in a specific direction, and a weak response when the stimulus moves in the opposite direction. Without further high-level processing, these responses are utilised to drive the optomotor control which serves to stabilise the insect's orientation with respect to the visual environment. Thus, an essential requisite of visually guided behaviour in insects is the ability to detect and use visual motion information. This ability, in particular, with a simple but efficient neural network as a source of biological inspiration for constructing real-time vision systems, made insect vision especially attractive to machine vision scientists.

1.1.2 Neuromorphic Engineering and Smart Sensors

"*There is nothing that is done in the nervous system that we cannot emulate with electronics...*" (Mead 1990)

Even though a number of researchers have been attracted by insect vision in designing real-time vision systems (Brooks 1986; Pichon et al. 1989; Horridge 1992; Osorio & Sobey 1992; Franceschini et al. 1992), these systems are often implemented in a classical way in which a separate imager and processor are combined, and available off-the-shelf components, such as charge-coupled device (CCD) cameras and digital computers, are usually employed. This implementation has faced serious problems of data communication bandwidth and storage requirements between components (Poggio et al. 1988; Koch 1989).

In fact, when it comes to information processing problems, the brain of a housefly performs the computation far better than state-of-the-art digital computers. For many problems, especially those which are ill-conditioned such as vision problems, the biological nervous system's performance is many orders of magnitude more efficient than that of digital com-

puters. For example, statistics show that 1kW is consumed to operate a super-computer which weighs 10^5g, while a fly brain, which weighs less than 1mg, utilises only 1μW (Götz 1987; Sejnowski & Churchland 1992). This advantage results principally from a highly parallel architecture, the use of elementary physical phenomena as computational primitives, and the representation of information by the relative values of analog signals rather than the absolute values of digital signals (Mead 1989a, 1990). This observation has motivated the hope of mimicking biological neural systems into information processing machines using analog Very Large Scale Integrated (VLSI) circuits rather than their digital counterparts. Experimental studies have shown that some of the physical properties of analog CMOS (Complementary Metal Oxide Silicon) exhibit similarities to some phenomena governed by the electrical behaviour of neurons and neural systems (Mead 1990). Fortunately, over the past ten years, tremendous progress in the construction and demonstration of analog VLSI information processing systems has been made. For instance, early works in analog VLSI have led to several implementations of sensory systems (Tanner & Mead 1984; Sivilotti et al. 1987; Mead 1989; Koch 1989; Mahowald 1991; Moore & Koch 1991). The rapid development of analog VLSI circuit technology has provided us a medium in which it is currently possible to design and fabricate artificial neural systems such as vision systems, whose architecture and design principles are based on those of biological nervous systems. This new field is referred to as *Neuromorphic Engineering,* and was introduced by Carver Mead (1990).

An ultimate goal of the Neuromorphic Engineering field is to obtain a maximum energy efficiency of analog information processing systems. Biological nervous systems are more efficient in power usage than our present modern digital technology by a factor of 7 to 9 orders of magnitude (Mead 1990; Sejnowski & Churchland 1992). Such a disproportion has two main causes. Firstly, most of the energy is dissipated when transistors switch from one state to another. Secondly, the energy is also used to charge up the parasitic capacitance associated with a node, which usually is wire capacitance (Mead 1990; Sejnowski & Churchland 1992). Early achievements in this field include Mead's well-known *Silicon Retina*, a device which computes the spatial and temporal derivative of an image projected onto its 2D array of photoreceptors. The device uses a hexagonal resistive layer which performs as the simplest possible discrete form of the Laplacian differential operator (Sivilotti et al. 1987; Mead & Mahowald 1988; Mahowald & Mead 1991). Compared with other conventional vision systems, the silicon retina is a paragon of efficiency in terms of performance, power consumption, and compactness.

Neuromorphic Engineering has also initiated a so-called *smart sensor* paradigm in which most of the signal processing is incorporated into the sensor and its associated circuitry in order to reduce transmission bandwidth, storage requirements, and subsequent stages

of computation (Koch 1989; Åström 1993; Bouzerdoum et al. 1994; Nguyen et al. 1993a). This implementation framework is starting to emerge as a possible competitor to more general-purpose digital vision systems. However, the signal processing is very often occurring in the digital domain rather than the analog domain. A possible solution is, of course, a mixed analog-digital design framework, on which the work in this research has been founded.

1.1.3 Summary

Insect vision has been chosen as a model for constructing *smart micro-sensors* in this research because insects can see better than machines and many fascinating lessons that we can learn from insect vision can be applied to build machine vision systems. Insect vision offers a low-level but efficient mechanism of visual processing for detecting and utilising the information of directional motion. If motion is successfully detected, the resulting information can potentially be utilised in many real-time applications. In summary, the primary motivation is the desire to apply insect vision principles to real-world problems.

The mixed analog-digital VLSI technology has been selected as a main technology in designing two *smart VLSI micro-sensors* since this technology can provide a medium in which a complete vision system combining the sensor and its associated processing elements can be designed. The first, called *smart micro-sensor I,* is a concept demonstrator, while the second, called *smart micro-sensor II*, is a test facilitator.

1.2 Research Methodology

The first work on the methodology of vision research appeared in the work of David Marr and his colleagues from Massachusetts Institute of Technology (Marr 1982). Marr argued that if one hopes to achieve a full understanding of a complex system, such as an information processing system, as complicated as a nervous system or even a large computer program, then one must be prepared to contemplate different kinds of explanation at different levels of description that are linked, at least in principle. Marr carefully proposed a methodology, which has eventually been known as *Marr methodology*, and which is composed of three levels as shown in Table 1.1. These levels can be briefly explained as follows.

A. Computational Theory

The level of computational theory is the first as well as the most important level of the whole process because the computations that underlie perception depend more upon the computational problems that have to be solved than upon the particular hardware in which their solutions are implemented (Marr 1982). In other words, it is a level of explanation.

Computational theory	Representation and Algorithm	Implementation
Defining the information processing problem, whose goal characterizes the computational theory. Defining the abstract properties of the computation. Discovering the properties of the visual world that constrain the computational problem	Investigating the data representation and developing the algorithms which can be used to operate the studied computation.	Implementing the algorithm into hardware. Studying architectural issues of the designed engine.

Table 1.1 The three levels at which an information processing system must be understood (adapted from Poggio 1981).

Thus, it is worth to develop, through strongly mathematical means, the relationship between the quantity to be computed and the observations.

B. Representation and Algorithm

The second level of the Marr methodology involves representing and constructing algorithms for processing information obtained from the study in the first level. Based on the nature of the computational theory, the information should be constructed in an appropriate form which may then allow one to design suitable algorithms. There is usually a wide variety of representation of the information. So, this issue is very important because the way that the information (obtained from the study of the computational theory) is structured can greatly affect how easy different things can be done with their structure. That is the data structure greatly affect the way that the information is processed. Thus, which representation is selected will usually depend upon any particularly desirable or undesirable properties the algorithms may have; for instance, one algorithm may be much efficient than another, or another may be slightly less efficient but more robust (Marr 1982). Whatever the way the information acquired from the first level is manipulated to obtain a suitable data structure and an algorithm is, the second level can be considered as the bridge connecting the first and third level of this methodology.

C. Implementation

The level of implementation is the last level of the whole process. Physical realization is also a challenging task which makes the theory and resulting algorithm to be mapped into hardware and tested in real-time. This level concerns not only the functional correctness of the theory and algorithm but also the robustness of any resulting system in real-time. There are some techniques which are very elegant in their mathematic formulations. However,

their real-time operation is not appropriate due to either susceptibility to noise or computational burden. Thus, it can be said that the purpose of the last level is to evaluate the level of success of the theory and algorithm.

Although the Marr methodology was proposed in the framework of the general-purpose vision in the computer vision community (Marr1982), this methodology was recently re-evaluated, and its generality, which makes it widely applicable not only in the restricted field of machine vision but also in the general analysis of information-processing systems, was also exposed (Jolion 1994; Mead 1990). Thus, it is not surprising that the Marr methodology is also successfully adopted as a standard for the process of designing the two *smart VLSI micro-sensors*.

Fig. 1.1 shows the strategy of the multi-disciplinary research presented in this thesis. Works in different fields which are involved in this research are divided into three different modules corresponding to the three levels in the Marr methodology. The first module, which corresponds to the level of computation theory, is the study in insect vision from which different neural models were proposed. The second module, which reflects the level of representation and algorithm, is involved in the field of image processing, where feature representation of motion trajectories and different tracking algorithms are proposed. Finally, the third module, which is equivalent to the level of implementation in the Marr methodol-

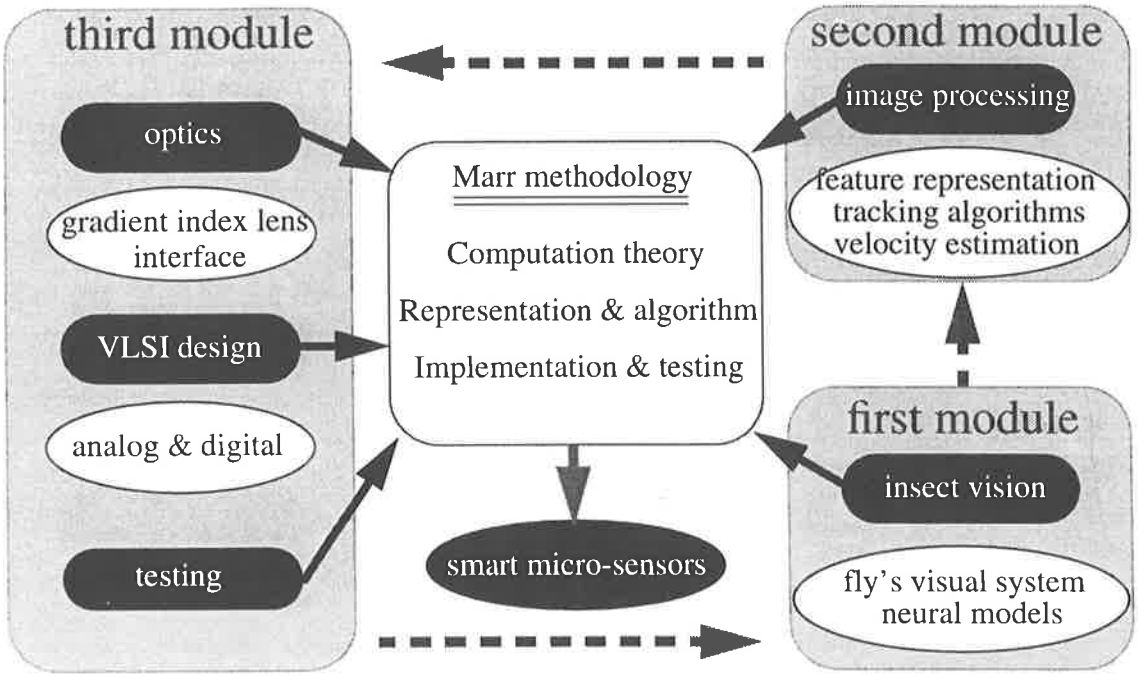


Fig. 1.1 Research strategy. Works in different fields are divided into three different modules under the umbrella of the Marr methodology.

ogy, shows works in hardware implementation of a smart vision system. The works in this module include the VLSI design and the optical interface of the system, and of course, the testing of the system in real time.

The Marr methodology can be directly applied in either a top-down or bottom-up fashion as classified in Aloimonos & Shulman (1989). The top-down approach corresponds to starting with the computational theory and ending up with either a software or a hardware vision system to testify the theory; whereas the bottom-up approach starts with building up a vision system and then attempts to formulate an appropriate theory. Either way of exercising the Marr methodology, only one design cycle is performed. However, satisfactory products of many problems in real life may be achieved after not one design cycle, but rather a few cycles. Any advantages or drawbacks learned in early cycles are useful for the study, and for improving the design in later cycles. The fashion of repeating a same design cycle for a number of times is referred to as the *iterative design approach,* which has been applied in other disciplines such as Software Engineering (Ghezzi et al. 1991), and VLSI design (Elmasry 1985). In fact, the Marr methodology has been iteratively exercised in this research in order to achieve a system that successfully works in real time. The achievement of the two designs of the two smart micro-sensors reflects two design cycles which have been carried out so far.

1.3 A Guided Tour of the Thesis

With the exception of Chapter 1 and Chapter 9, which serve as the opening and ending, this thesis is logically divided into two main parts: *Analysis* and *Implementation*. This structure is illustrated in Fig. 1.2.

The first part, namely Analysis, which consists of Chapters 2 through 6, contains basic material dealing with the first two modules of the research strategy as shown in Fig. 1.1. Chapters 2, 3, and 4 deal with matters of the first module, which introduce the reader to the state of the art of visual motion computation, and more importantly to the computational theory of different neural models for motion detection based upon insect vision. The work in Chapters 5 and 6 reflects the second module, which concerns two different schemes for feature representation of motion trajectories and tracking algorithms for velocity estimation.

The second part is named Implementation because it contains material dealing with the third module of the research strategy. It comprises Chapters 7 and 8, which describe two different implementations of the two smart micro-sensors and real-time tests of the designed sensors. Chapter 7 deals with the design of the smart micro-sensor I, and Chapter 8 expresses the smart micro-sensor II.

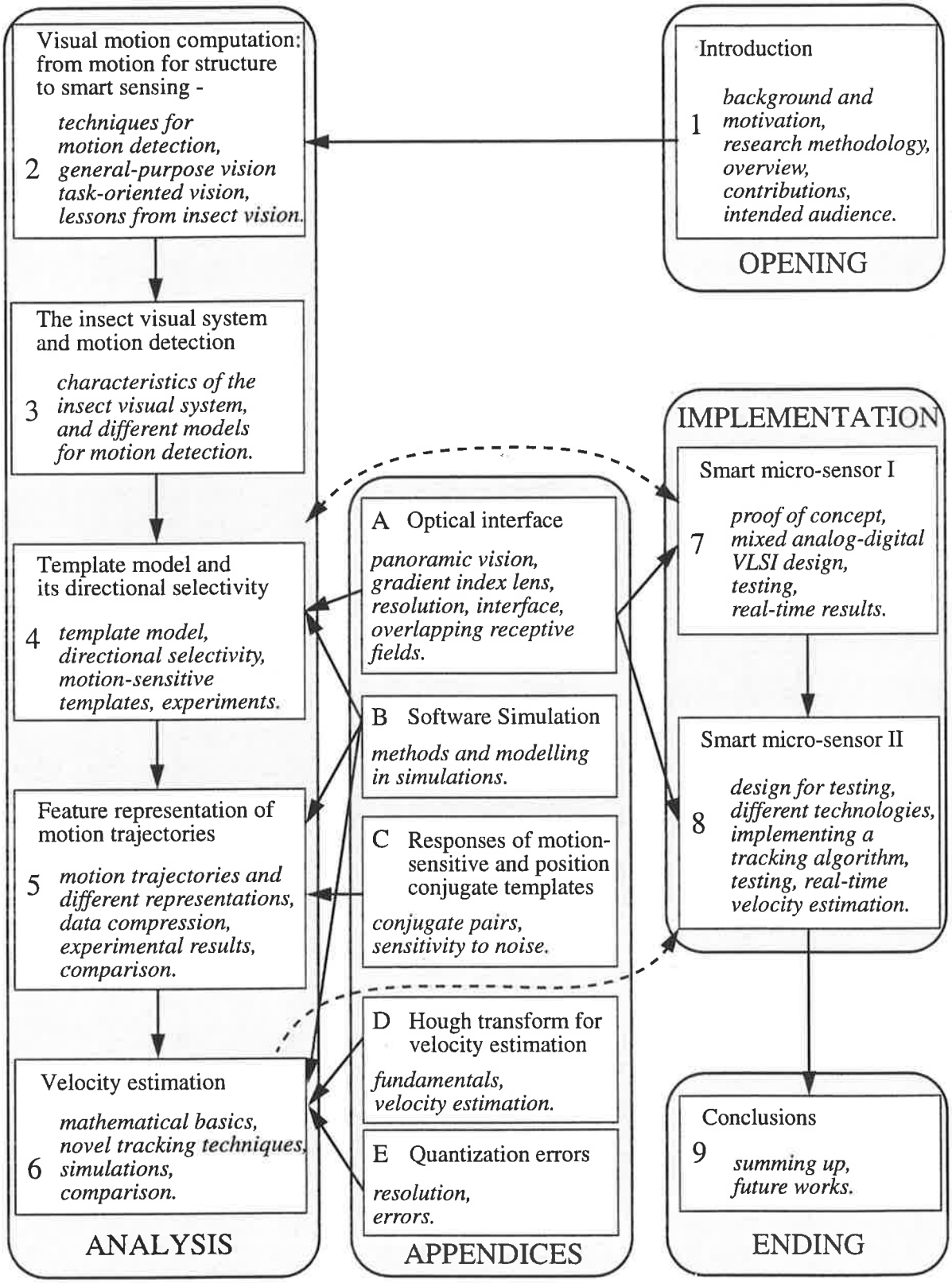


Fig. 1.2 Thesis organisation.

However, in order to assure the validity of the analysis work in Chapters 4 and 5, some of the experimental results obtained from testing the smart micro-sensor II are early presented in those chapters.

Furthermore, Appendix A, which provides the optical interface of the smart micro-sensors, complements Chapters 4, 7, and 8. Appendix B describes the procedure of software simulation whose results are used for works in Chapters 5 and 6. Appendix C, which supports Chapter 5, provides simulation results of template responses. Appendices D and E complement Chapter 6. Appendix D provides mathematical fundamentals of Hough transform and velocity estimation using Hough transform, while Appendix E provides mathematical basics on quantization errors.

Although the reader can follow Chapters 1 through 9 sequentially, the actual research process has been carried out as shown by directional arcs in Fig. 1.2. A directional arc between two chapters indicates that the source chapter, e.g. Chapter 2, can be considered as the base for the destination chapter, e.g. Chapter 3. The bidirectional link between Chapters 4 and 7 illustrates that the smart micro-sensor I had been designed before further analysis work in Chapters 4 (partly), 5 and 6 were carried out. However, lessons which were learned from the first design helped to proceed to the analysis work in those chapters, which were then applied in the second design presented in Chapter 8. Thus, the process of accomplishing the work in Chapters 4 (partly) and 7 can be considered as the first iteration of the Marr methodology, while that in Chapters 4 (partly), 5, 6, and 8 belongs to the second iteration.

1.4 Contributions

The designs of the two smart micro-sensors have been carried out by the *Bugeye* group at the University of Adelaide, so it is unavoidable that different members of the group have contributed in one way or another to these designs. In particular, the VLSI design of the smart micro-sensor I was mainly done by Moini (1994) and Yakovleff (1995), and the analog VLSI front-end processor of the smart micro-sensor II was designed entirely by Moini (Moini et al. 1995). However, for the sake of integrity, I have decided to present these designs in some details in this thesis.

The principal contributions given by the author are as follows:

- A critical review of visual motion computation - state of the art (Chapter 2).
- A review of insect vision, principles of motion detection, and two neural models for motion detection: the correlation model (Reichardt 1961) and the shunting lateral inhibition (Bouzerdoum 1991) (Chapter 3).

- The investigation of the optical interface of the sensors (Appendix A).
- The development of a mathematical formulation of directional selectivity of the template model, which aids in showing that the template model is functionally similar to the correlation model (Chapter 4).
- A classification of template responses to motion using the proposed formulation and a heuristic investigation (Chapter 4).
- The study of two different schemes for feature representation of motion trajectories using template responses (Chapter 5).
- The development of two tracking algorithms for velocity estimation, stair-step and backward tracking algorithms, which correspond to the two afore-mentioned representation schemes (Chapter 6).
- A comparison among the two proposed tracking algorithms and two others algorithms proposed by Sobey & Horridge (1990) and by Yakovleff (1995) (Chapter 6).
- Real-time tests of the smart micro-sensor I to different moving stimuli (Chapter 7).
- The design of the template and velocity estimation processors for the smart micro-sensor II (Chapter 8).
- Real-time velocity measurements using the smart micro-sensor II (Chapter 8).

1.5 Intended Audience

It may already have become apparent that the intended audience reader is expected to be generally abreast with insect vision principles for motion detection, Marr methodology, and VLSI design. The reader should be familiar with other computational techniques for computing visual motion and different approaches for solving real-time problems that are related to visual motion, which are intensively provided as a review in Chapter 2.

It is also assumed that the reader has some background knowledge of certain functional aspects of insect vision, particularly neuronal mechanisms for motion perception. Some knowledge in high-level visual processing, such as Hough Transform and its applications, data compression, is assumed to be familiar to the reader.

Some general understanding of VLSI system-related aspects, such as architectural issues, different integrated circuit (IC) technologies, design, are also assumed to be known by the reader although an in-depth knowledge of VLSI design methods is not assumed.

PART 1: ANALYSIS

Chapter 2

Visual Motion Computation - State of The Art

2.1 Introduction

Motion creates an optical flow field that contains information useful for a variety of visual tasks including the recovery of environmental structure, reconstruction of the third dimension, delineation of surface boundaries, visual kinesthesis, control of eye movement, mediation of size constancy, pattern vision, collision avoidance, navigation and so forth; see (Nakayama 1985) for a review. It is, thus, not surprising that motion detection has received a great deal of attention on the part of researchers from different disciplines such as neurophysiology (Barlow & Levick 1965; Michael 1968), psychology (Gibson 1950; Cutting 1986), and computer science (Ullman 1979; Longuet-Higgins & Prazdny 1980; Bruss & Horn 1983; Tsai & Huang 1984; Singh 1992; Jähne 1993).

The state of the art in visual motion computation has been undergoing an evolutionary process. During the last 15 years, Marr's philosophy of vision as a process for re-creating the three-dimensional (3D) structure of the world was intensely used as a guide in the field of computer vision (Marr 1982). Motion for structure has been considered a part of this philosophy since visual motion can be used as a cue in recovering the environmental structure (Ullman 1979a, 1979b; Marr 1982). Within this paradigm, the problem of visual motion is carried out at two independent stages: the measurement of the optical flow field, and the recovery of the scene description from the optical flow information. As a result, numerous techniques for optical flow computation have been developed; for instance, see Bouzerdoum & Pinter (1991), Vega-Riveros & Jabbour (1989), and Aggarwal & Nandhakumar (1988) for reviews of those techniques. However, these techniques suffer from two primary problems: computational bottleneck and robustness. These problems hamper real-time applications of the existing theories on visual motion computation.

Furthermore, in some applications such as controlling the navigation of mobile robots, a complete representation of the physical world is not necessary. In other words, a number of vision-related problems can be solved directly from specifications without the structure-recovering stage. This key idea leads to a new paradigm referred to as *task-oriented vision*

or *purposive vision*.

There still exists, however, on-going controversy between the general-purpose (recovery paradigm) and task-oriented philosophies (Tarr & Black 1994a, 1994b; Aloimonos 1994; Jain 1994; Brown 1994; Edelman 1994; Tsotsos 1994; Fischler 1994; Aggarwal & Martin 1994; Christensen & Madsen 1994; Sandini & Grosso 1994). Tarr & Black (1994a, 1994b), inspired by human's multi-functional vision system, claimed that the recovery paradigm is viable, and moreover, provided a promising framework for understanding and modelling general purpose vision in humans and machines. In contrast, Aloimonos (1994), influenced by insect vision systems, suggested that vision has a purpose, and that purpose is action. Consequently, the construction of a vision system should be dependent on the system's purpose or task. This purposive vision paradigm has been favoured for solving the problems faced by the motion for structure paradigm (Aloimonos 1990). This task-oriented approach highlights the necessity for a design which identifies the need for simple visual processing mechanism suited to a particularly specific problem. An introduction to this purposive vision was proposed by Bajcsy (1987) who stated that "*we do not just see, we look.*", thus emphasizing the role of vision in controlling our actions toward the physical world in order to meet a particular goal. Hence, the purpose of vision should not be reduced to the description of a scene, which is supposed to be done at the high processing levels. A particular case of the purposive vision is *smart sensing* which may be defined as the "*selective, task-oriented gathering of information from the visual world*" (Burt 1987). This approach is associated with early vision tasks, as distinct from general vision tasks of describing the world, and also takes into account the constraints of hardware implementations. See Jolion (1994) for a review of computer vision methodologies.

Parallel to the perceptual transition from the general-purpose to task-oriented methodologies in the computer vision community, biological visual systems have vigorously inspired a number of computer vision researchers (Brooks 1986; Pichon et al. 1989; Horridge 1992; Osorio & Sobey 1992; Franceschini et al. 1992; Bouzerdoum & Pinter 1992; Bouzerdoum 1993). They have found that one of the most task-oriented visual systems in nature is that used by insects, which have not evolved much beyond the ability to separate the direction of a discontinuity in the visual flow field. It has been shown experimentally that insects detect movement between neighbouring points of the sampling lattice of the compound eye by making use of elementary movement detectors in the initial stages; see Buchner (1976, 1984) and Franceschini et al. (1989) for instance. Although their visual system is very simple, insects survive well in their natural habitat. They rely on their sense of sight to follow a mate, avoid potential dangers, or land on a target. The insect visual system is now an inspiration for machine vision scientists.

In this chapter, we briefly summarize the state of the art of visual motion computation from *motion for structure* to *task-oriented and insect-inspired vision* in an evolutionary fashion. Section 2.2 briefly introduces the *motion for structure* paradigm which was distinctly expressed in Marr's philosophy on vision; it also presents a number of algorithms for motion measurement within this paradigm and discusses a number of limitations of their practical performance. Section 2.3 describes an alternative view, *task-oriented vision*, while Section 2.4 puts forward the influence of insect vision on the practical design of machine vision systems. Finally, the chapter ends with a summary.

2.2 *Motion for structure* and Motion Measurement

More than ten years ago, in the book *Vision* (Marr 1982), which emphasises the desire to emulate human vision, the author stated that "*Vision is the process that creates, given a set of images, a complete and accurate representation of the scene and its properties*". This was Marr's answer to the question: What is vision? In other words, Marr's hypothesis was that we can investigate the principles of visual perception by considering the purpose of vision to describe environmental structures. This approach is known as the *recovery school* (Aloimonos & Rosenfeld 1992). Following Marr's hypothesis, we should consider the task of vision as being the construction of a detailed representation of the physical world, independent of tasks under consideration. This viewpoint is prevalent for example in Chellappa & Kashyap (1992) who stated that "*The goal of an image understanding system is to transform two-dimensional spatial data into a description of the three-dimensional spatio-temporal world... [such a system] infer 3D surfaces, volumes, boundaries, shadows, occlusion, depth, colour, motion,...*" This hypothesis has been widely followed and visual motion has long been used as a cue to recover the environmental structure which has led to the paradigm *motion for structure* (Ullman 1979a, 1979b; Marr 1982; Tsai & Huang 1984). Within such a paradigm, the problem of visual motion splits naturally into two separate two stages - the measurement of image motion or optical flow field (Horn & Schunck 1980; Hildreth 1984a, 1984b), and the recovery of scene description from optical flow information (Longuet-Higgins & Prazdny 1980; Sundareswaran 1992). Although these two stages can be treated independently, the performance of the second stage depends upon the availability and accuracy of information from the first stage (Barron et al. 1990).

To keep this chapter close to the context of the thesis, which emphasises motion detection, only the first stage of the general visual problem, optical flow computation, is briefly reviewed. It is very difficult to produce fair and conclusive comparisons among techniques for computing optical flow, and hence such comparisons are not the aim of the following discussion. However, comparative reviews of techniques for computing optical flow can be

found in the literature (Singh 1992; Barron et al. 1994). Other important works related to visual motion are not discussed because they are of less relevance to the objective of this thesis. For example, for those who are interested in works in the second stage, that is, the scene recovery which is less relevant to the purpose of this thesis, see Huang & Netravali (1994), Ullman (1979b), or Bolles et al. (1987). Although there are a number of techniques proposed for the estimation of the optical flow field, four categories of approaches have been pursued in the past: Gradient Models, Region-based Matching Methods, Spatio-Temporal Energy Models, and Phase-based Models.

2.2.1 Gradient Models

The optical flow field, or the velocity field that represents the motion of objects across an image, can be calculated from local spatial and temporal derivatives of the image intensity function. The gradient techniques are based upon a common assumption of *conservation of image intensity*, that is, the observed intensity of image elements on the image plane is constant over time. Any changes in intensity are assumed to be caused by translation of the local image intensity and not by changes in lighting, reflectance, and so forth. Some techniques use a function of image brightness, such as spatial gradient of image intensity (Verri et al. 1990), rather than the image intensity itself. However, the underlying principle of computation remains the same (Buxton & Buxton 1984).

Based upon the brightness constancy assumption as a local constraint, the gradient constraint is formally stated in Eq. (2.1), whose detailed derivation was given by Horn & Schunck (1981). See also (Schunck 1985, 1986).

$$\frac{dI}{dt} = \frac{\delta I}{\delta x}\frac{\delta x}{\delta t} + \frac{\delta I}{\delta y}\frac{\delta y}{\delta t} + \frac{\delta I}{\delta t} = 0 \tag{2.1}$$

or by rearranging this equation we have

$$\nabla\vec{I} \cdot \vec{V} = -\frac{\delta I}{\delta t} \tag{2.2}$$

This equation, known as the brightness constancy equation or the image flow constraint equation, relates the temporal derivative of the intensity function $\delta I/\delta t$ at a point, and the spatial gradient $\nabla\vec{I}$ in the neighbourhood of the point to the instantaneous velocity $\vec{V}$ at that point in the image. Thus, the local constraints provide one linear equation in the two variables of the velocity vector $\vec{V} = (u, v)$. Consequently, the velocity vector cannot be calculated locally without applying additional constraints. This problem is referred to as the *aperture problem*, which arises when a straight edge is viewed through a small aperture

(Ullman 1979a; Marr & Ullman 1981). All that can be obtained directly from this equation is the normal velocity, V_n, that is parallel to the intensity gradient vector as

$$V_n = -\frac{\delta I/\delta t}{\left|\vec{\nabla I}\right|}. \tag{2.3}$$

Thus, the problem of optical flow computation from Eq. (2.2) is not a well-posed problem in the sense of Hadanard (Poggio & Kock 1985; Bertero et al. 1988) since Eq. (2.2) does not have a unique solution for $\vec{V}$. To solve this problem, and hence to recover both components of the velocity vector, at least one additional constraint must be imposed. The most commonly used constraint is the smoothness of the optical flow field (Horn & Schunck 1981; Hildreth 1984a; Nagel & Eckelmann 1986; Nesi 1993). That is, if the optical flow field smoothly changes over the image plane, and if the intensity gradient vector locally varies its direction within a small area of an observed point, the other component of the velocity vector can also be determined. Such an estimate will, of course, be approximate, but may be the best possible under some circumstances.

Horn & Schunck (1981) highlighted the ill-posed nature of the problem and developed a variational formulation for estimating the smoothest velocity field. Firstly, they postulated that the rate of change of image intensity, Eq. (2.1), is not perfectly zero because of noise and quantization errors. They also used the assumption of *smoothness of optical flow field* to impose the additional constraint necessary for solving the aperture problem. This constraint derives a term called the smoothness error, which is then combined with the quantization error to form a total error. The velocity can be computed by minimizing this total error over the entire image.

Hildreth (1984a, 1984b) proposed a technique that bounds the smoothness requirement to be exercised only along image contours †, instead of smoothing out the velocity field over the entire image. The image contours are firstly detected by applying the zero-crossing edge detection technique to the input image (Marr & Hildreth 1980). Secondly, the smoothness error is computed along the image contours. Thus, the technique gives the velocity field only along the contours, not everywhere in the image.

Nagel (1983, 1986) developed a technique using second-order derivatives to compute optical flow. Nagel argued that the isotropic manner of the smoothness error proposed by Horn & Schunck is a major drawback in the sense that it blurs the optical flow uniformly in all directions. Nagel proposed a so-called *oriented-smoothness* constraint in which smoothness is not necessarily enforced in the regions with strong second-order intensity variations

†. An image contour consists of points where discontinuities in gray-scale occur.

(such as corners). In addition, this constraint is strongly imposed along the image contour, whereas it is weakly applied across the contour in the neighbourhood of edges.

Nesi (1993) proposed a new technique based on the theory of variational convergence to compute the optical flow field. He recognised the major difficulty of the techniques discussed so far is the presence of propagation effects which produce the loss of the information associated with discontinuities in the image intensity. This is due to the smoothness assumption being violated around discontinuities in the image plane, which correspond to movement boundaries and surface boundaries. By movement boundaries we refer to boundaries of coherently moving objects. Such boundaries always take place along surface boundaries which in turn are not always the movement boundaries. There is a way to overcome this difficulty by taking into account the presence of discontinuities. In particular, Nesi took into account this factor by including it in his proposed smoothness error function. This technique provides a solution for the measurement of smooth optical flow which maintains the information associated with discontinuities.

2.2.2 Region-based Matching Models

Rather than being based upon the gradient of the image intensity, which is usually not robust against noise, the region-based methods are based on the image intensity itself. Such methods define the velocity $\vec{V}$ at the location as the shift $D = (d_x, d_y)$ that yields the best fit between image regions at different times. Searching for the best match amounts to maximizing a similarity measure between patterns (over D) using methods such as the normalized cross-correlation, or minimizing a mismatch measure using, for example, the sum-of-squared difference (SSD):

$$SSD_{1,2}(X;D) = \sum_{j=-n}^{n} \sum_{i=-n}^{n} W(i,j)\,[I_1(X+(i,j)) - I_2(X+D+(i,j))]^2 \qquad (2.4)$$

In Eq. (2.4), $I_1(X)$ and $I_2(X)$ represent the pixel intensities at the location $X = (x, y)$ in the first and the second images respectively; $W(X)$ denotes a discrete 2D function, which is simply defined as a pattern around that location to be matched between two images, and $D = (d_x, d_y)$ has integer values.

Anandan (1989) proposed a coarse-to-fine SSD-based matching algorithm which uses a Laplacian pyramid architecture. In fact, the Laplacian pyramid is very useful in decomposing each input image into a set of images containing information at different scales. Hence, it enables the determination of large displacement between images and helps to enhance image structure. The best match is selected as the SSD reaches its minimum. This distribution of SSD values is referred to as the SSD surface over the search area. Furthermore,

the principal curvatures of the SSD surface at the best match location are calculated with their perspective directions, which are then used to obtain the maximum and minimum values of so-called *matching confidence*. Finally, the smoothness constraint is also employed to find an optimum solution of the image velocity.

Singh (1990, 1992) implemented a two-stage matching method in which the first-stage consists of directly estimating local velocities from intensity data. Singh extended the work of Anandan to handle the case of multiple possible matches more efficiently. Furthermore, Singh proved that this technique can be considered as a generalised framework from which other techniques can be derived.

In the first stage, the SDD distribution is determined using multi-images, also known as multiframes (not limited to two images only). The easiest multiframe-case involves three images. A purpose of using multi-images is to smooth out the spurious SSD minima caused by noise or periodic texture. Furthermore, the SSD is then transformed into a probability distribution which is then used to determine a subpixel velocity. *Confidence* measures are also obtained for this velocity.

In the second stage, the velocity is propagated using neighbourhood information. An optimal velocity is obtained by employing the smoothness constraint, that is, the velocity is determined by minimizing a total error function.

Kouzani (1995) proposed an approach which uses fuzzy set theoretic principles to perform matching. Furthermore, this approach takes into account the historical values of motion direction and of velocity to estimate the final velocity (Kouzani et al. 1995; Bouzerdoum et al. 1995). Thus, this approach effectively uses multiframes rather than two frames.

2.2.3 Spatio-temporal Energy Models

Optical flow computation techniques in this category were typically derived from the analysis of visual motion in the frequency (Fourier) domain. These techniques extracted velocity information by designing velocity-tuned filters which could fit spatio-temporal energy output to a plane in the frequency space. To illustrate this, consider a 2D intensity image $I_0(x, y)$ moving with a constant velocity $\vec{V} = (u, v)$, that is, the spatio-temporal intensity distribution takes the form

$$I(x, y, t) = I_0(x - ut, y - vt) \tag{2.5}$$

The 3D Fourier transform of this function is connected to that of $I_0(x, y)$ as

$$I(f_x, f_y, f_t) = I(f_x, f_y)\,\delta_p(f_t + uf_x + vf_y) \tag{2.6}$$

where δ_p is the plane impulse function which is zero everywhere except on the plane:

$$f_t + uf_x + vf_y = 0 \tag{2.7}$$

Therefore, the spectrum of a moving image lies in an oblique plane through the origin. The orientation of this plane determines the speed and direction of motion.

Heeger (1987a, 1987b) developed a technique using spatio-temporally orient-tuned filters to extract optical flow. The technique is formulated as a least-squares fit of the spatio-temporal energy to a plane in the frequency domain as described in Eq. (2.7). A set of 12 Gabor-energy filters is used, all of which are tuned to the same spatial-frequency band but to different spatial orientations and temporal frequencies. Heeger determined the sum of measured and predicted energies of the set of filters. A least-squares method is then employed to obtain a velocity estimate by minimizing the difference between the predicted and measured motion energies. Heeger's approach assumes that images are textured, thereby it avoids the need of a smoothness constraint. However, it is computationally very intensive.

Huang & Chen (1995) pointed out that most of the frequency-based methods (Watson & Ahumada Jr. 1985; Heeger 1987) which are based on the Gabor-energy filter are computationally intensive since these filters are not separable in space or time. Since the Gabor-type filter can only have certain directional pairs of filters, these approaches have the orientation selection problem. In order to ameliorate the effects of the problems, Huang & Chen proposed a different method using 3D steerable filters which are Gaussian filters rotated to a defined direction in the frequency domain. The oriented energy in that direction is measured.

To bypass the computational cost of 3D inseparable Gabor-typed filters (Tanaka & Shirai 1994) this steerable filter is implemented as a combination of three basic functions which are the first derivatives of the Gaussian in the *x*-, *y*-, and *t*- directions respectively. Based on Parseval's theorem the total oriented energy can be determined as the sum of partially oriented energies in the *x*-, *y*-, and *t*- directions. Finally, the velocity is estimated by minimizing the total oriented energy.

2.2.4 Phase-Based Models

The techniques in this category estimate velocity by analysing the phase behaviour of bandpass filter outputs. The band-pass filters are used to decompose the input signal according to scale, speed, and orientation. The output of such a filter may be expressed in a complex form as

$$R(x, t) = \rho(x, y, t)\, e^{i\phi(x, y, t)} \tag{2.8}$$

where $\rho(x, y, t)$ and $\phi(x, y, t)$ are the amplitude and phase parts of the filter output $R(x, y, t)$. The component of 2D velocity in the direction normal to level phase contour † is determined as $\vec{V}_n = s\vec{n}$ where the normal direction, $\vec{n}$, and speed, s, are calculated as

$$\vec{n} = \frac{-\phi_t(x, y, t)}{\|\nabla\phi(x, y, t)\|}, \text{ and } s = \frac{\nabla\phi(x, y, t)}{\|\nabla\phi(x, y, t)\|} \tag{2.9}$$

where $\nabla\phi(x, y, t) = (\phi_x(x, y, t), \phi_y(x, y, t))^T$. The phase derivatives are evaluated using the identity

$$\phi_k(x, y, t) = \frac{Im\left[R_k^*(x, y, t)\, R_k(x, y, t)\right]}{|R(x, y, t)|^2}, \text{ and } k = x, y, t \tag{2.10}$$

where R_k^* is the complex conjugate of R_k.

Fleet & Jepson (1990) were the first who generalised the use of phase information for estimating optical flow. Actually, this technique is similar to the differential technique, but is applied to the phase rather than intensity function, which makes it less sensitive to non-motion changes in intensity, and non-translational motion. Finally, a linear velocity model can be used to fit to each local region provided with the normal velocity components (Barron et al. 1994).

2.2.5 Limitations in Real-Time Applications

The general objective of all the previously discussed techniques is to compute optical flow as accurately as possible in order to minimise errors in the subsequent stage of recovering the environmental structure. On the whole, these techniques have been based upon the assumption that changes in the image brightness are induced by motion only. This assumption often results in an ill-posed problem which cannot be solved without an additional constraint. A frequently used constraint is the smoothness of optical flow that leads to an optimization problem in which velocity is selected so that an associated functional is minimized. Many researchers have been keen on this approach because it is well supported by the mathematical tool called *regularization theory* (Poggio & Koch 1985; Poggio et al. 1985; Bertero et al. 1988). However, the existing techniques are far from being useful in real-time applications due to two major difficulties which are discussed below. Furthermore, the necessity of 3D representations of the environment is also discussed to clarify its role in

†. A phase contour is defined similarly to the image contour, but it is found in a phase image instead of an intensity image.

the computer vision world.

A. Computational Bottleneck

The problem of minimizing an associated functional mentioned above can be solved by a relaxation algorithm which involves a numerical iterative method. Firstly, the problem can be transformed into a set of differential equations by applying Euler-Lagrange equations, which are subsequently discretized and expressed as a sparse linear system. It usually takes a long time, perhaps a few hundreds or thousands of iterations, to reach a final solution (Li & Wang 1993). If these techniques were to be used in real-time systems, such as mobile robots, the overall performance of the systems would be slowed down considerably. For example, Papanikolopoulos et al. (1993) described a quite sophisticated vision system whose main image processing unit was equipped with two 20 Mflop floating point boards for performing all the computation load of the optical flow calculations. When this vision system was used for a mobile robot as a visual tracking unit, the robot could operate at an average speed of 1.8 m/min. In some cases where fast reactions with the environment for navigation are needed, these techniques are not suitable. Furthermore, these techniques are deeply involved in vast amounts of data which must be processed. As a result, complex hardware must be employed to solve the problem (Ali & Dagless 1992).

B. Robustness against Noise

There is no known algorithm to date that can compute optical flow in natural scenes with high accuracy. The performance of most of the techniques has been recently studied by (Barron et al. 1994) and the results show angular errors † in the range of 0.03^{o} to 91.71^{o} (Table 4.1 in Barron et al. 1994). The errors are even amplified in the second stage of the visual motion problem because of the needs for numerical differentiation of optical flow (Longuet-Higgins & Prazdny 1980). For instance, it was observed that an error of about 1% in the optical flow input results in about 100% error in the output (3D motion) (Aloimonos 1994; Spetsakis & Aloimonos 1988). In other words, the visual motion problem is an ill-conditioned and unstable problem because small perturbations in the input generate large errors in the output.

---

†. An angular error is defined as the angle between the correct $\vec{v}_c$ and an estimate $\vec{v}_c$, that is

$$\psi_e = \operatorname{acos}\left(\vec{v}_c \cdot \vec{v}_e\right).$$

A relative error of 10% is corresponding to an angular error of roughly 2.5^{o} when the speed of an object is near 1 image element per sampling interval. For more details see Barron et al. (1994).

C. Necessity of 3D Representations

If a visual system can robustly recover 3D description of the physical world, that is, if the system can take as its input a sequence of images and generate as output the 3D scene structure as well as the relevant motion parameters, then the system can solve all problems related to visual guidance. Thus, a behaviour agent (intelligent system) with such a complete visual system can possibly achieve any goal. However, an obvious question that arises is "Do we always need to build a 3D model of the world in order to achieve a particular goal?" The answer is "Of course, not always!" For example in the mobile-robotic world, the ability to avoid obstacles can be developed without a 3D model of scenes (Nelson & Aloimonos 1989). There is other evidence that motion detection can be directly used in mobile robots without 3D representations (Franceschini et al. 1992; Pichon et al. 1989; Brooks 1986, 1991). Of course, if information about the scene is available, it can be effectively utilized in such problems, but there are no robust and efficient techniques to date to rapidly obtain such information. Furthermore, if such a robust vision system were to be allocated to do such a simple task, it would be a waste of powerful resources. As described by Aloimonos & Rosenfeld (1991), "*If we can solve such problems directly, the structure from motion module is no longer needed.*"

2.3 Purposive Vision and Motion Detection

"*It is no longer possible to divide the process of seeing from that of understanding...*" (Zeki 1992)

Vision must be considered as a part of a complex behavioural system which may have perceptual, cognitive, and visuomotor control abilities. These abilities work in a cooperative fashion which strongly affects the system's hierarchical structure (Arbib & Liaw 1995). In nature, a visual system possesses a finite set of capabilities which allow it to tackle a number of tasks. In order to accomplish a task the system should intelligently extract and process certain information which is absolutely necessary in solving a given task from a vast amount of visual signals (Burt 1988; Dickmanns & Graefe 1988a). The capability of extracting useful information is related to the system's physiology.

In recent years, the realization has grown that the transfer of existing algorithms for motion measurement onto real-time applications has not been successful. This is not due to the techniques themselves, but perhaps due to the nature of visual motion perception as an aspect of the general task of reconstructing a 3D structure. Thus, there still exists a big gap between theory and real-time applications. To bridge this gap a number of researchers have tried to solve many vision problems directly with a limited amount of resources rather than

recover the 3D environmental structure first and then solve these vision problems (Nelson & Aloimonos 1989; Huang & Aloimonos 1994). This direct approach turns out to be a useful approach for the many real-time applications mentioned above; it is referred to as *task-oriented* or *purposive vision* (Aloimonos & Rosenfeld 1991a, 1991b; Aloimonos 1994; Jolion 1994). This approach highlights the necessity for a design which identifies the need for a simple visual processing mechanism suited to a particularly specific task. The more well-defined the task is, the better is the performance of the system. Within this perspective, vision is not defined as a process of creating dimensional descriptions, but is rather defined in term of its usefulness. In other words, rather than giving a precise definition of vision, we should answer the question "what is vision for?". An organism (or a robot) needs vision to fulfil some particular tasks. In the biological world, animals use vision to recognise friends, enemies, avoid danger, find food, and so on. Similarly, the vision required by visually-guided robots is dependent upon the environment in which they operate. Vision makes them capable of performing a variety of different tasks while interacting with their environment. It has been suggested that vision is most readily conceived in the context of the visual behavioural agents in which the visual system is engaged; these behavioural agents may not require detailed structural representations of the physical world (Ballard 1991; Brooks 1986).

In many problems related to visual motion, it has long been realised that motion should be either detected or measured depending upon applications. In some cases, it is sufficient to compute only certain properties of the optical flow field rather than measure it accurately and completely. For instance, to recover a 3D structural environment from motion, the velocity must be measured very precisely; however to avoid obstacles or collisions, motion should be detected but not necessarily measured (Ullman 1979a, 1979b, 1981; Prazdny 1980; Clocksin 1980). In the case of autonomous navigation a mobile robot must first detect all obstacles and then try to avoid them, but does not need to infer the exact shapes of all the objects surrounding it. The environmental structure is no longer needed. Motion detection could be based upon low-level information rather than being based on the structure of objects or recovery of 3D information of the world.

There are a number of models that are concerned with detecting motion only; these models were initially motivated from the study of insect vision (Reichardt 1961; Horridge 1990; Bouzerdoum & Pinter 1993; Beare et al. 1995). Details of these models will be described in the next two chapters. Behavioural and electrophysiological experiments indicate that motion detection is the primary characteristic of eyes and carried out by direction-selective neurons (Franceschini et al. 1989; Horridge & Marcelja 1990a, 1990b). By their visual responses, insects clearly exhibit that they "see", although the simplest examples have nothing that we could call object vision (Horridge 1987). A fascinating fact is that insects have very simple vision systems which have not evolved much beyond the ability to detect

objects and prevent collisions, but they marvellously accomplish a variety of tasks for surviving. Thus, insect vision has naturally given us a model from which we can construct machine vision systems which are dedicated to task-oriented vision. To this end, the clearest explanation of this approach is summarised as in Aloimonos (1994).

"The information contained in the visual signals is much more than the system actually needs or can cope with. This the fundamental problem of a vision system is determining what information from the image should be used and what representation of it needs to be built so that the relationship of the system with its world can best be implemented. In other words, the systems needs to recover partial information about the scene. Knowing what kind of partial recovery needs to be performed depends on the relationship of the system to its world; to put it differently, it depends on the tasks that the systems has to carry out, i.e., on its purpose"

2.4 From Insect Vision to Machine Vision

"*Animals can see better than machines so that their brains and behaviours are an inspiration for machine vision scientists*" (Osorio & Sobey 1992).

This statement expresses a new perspective in designing machine vision systems over the past few years. Features of the insect visual system can be mimicked into machine vision at several levels from optical, structural, to behavioural levels (Franceschini 1992; Horridge 1992; Osorio & Sobey 1992). Behavioural levels show the functional architecture, whereas the structural levels indicate the simple physiological architecture of insect vision. An early experimental work shows that insect vision can function with simple circuits that can be easily implemented in hardware (Pichon et al. 1989). Researchers' fascination with the marvel of insect vision has now progressed beyond the analysis stage to hardware implementations (Franceschini 1992; Nguyen et al. 1993a; Yakovleff et al. 1993).

2.4.1 Task-Oriented Vision

The task-oriented vision is the most important feature in insect vision in the sense that it provides a practical methodology to design machine visual systems. The development of eyes among different species has been evolving to different levels of complexity, depending upon living habitats and purposes for surviving, which can be to avoid danger, hunt for food, recognise a predator, and so forth. In particular, the visual systems of invertebrates and lower vertebrates do not necessarily have a complex nervous system (Horridge 1987). These visual systems have evolved to a level which is intermediate between simply detecting movements of contrasting edges and universal object discrimination. In fact, motion detection is the characteristic of these eyes. For example, a fly's visual system has not evolved far

beyond the ability to detect the direction of movement which is subsequently used for controlling the flight path (Bouzerdoum & Pinter 1992; Reichardt 1986, 1961). The complexity of the bee's visual system is no more than colour-blind motion detection and simple recognition ability (Zhang & Horridge 1992; Zhang et al. 1992; Srinivasan & Zhang 1993). Although the visual system of the bee is more complex than that of the fly, it is modularly constructed so that bees use different visual cues for different tasks (Lehrer 1994; Lehrer & Srinivasan 1994). These biological visual systems are essentially task-specific feature extraction visual systems, in which optical signals are intelligently filtered out from all the unnecessary information.

2.4.2 Simple Circuitry and Automatic Gain Control

Insect motion detection is primarily carried out by elementary motion detectors (EMDs) which basically consist of two asymmetric input channels whose interaction is non-linear (Borst & Egelhaaf 1989). The detectors have very simple circuitry which performs a non-linear interaction by a multiplication stage (Pichon et al. 1989; Franceschini 1992; Reichardt 1961). However, the real implementation of the detectors may be made even simpler without the multiplication stage, in which possible combinations of correct inputs are monitored instead of inspecting the output of the multiplication unit (Horridge 1990; Nguyen et al. 1993a; Yakovleff et al. 1993).

Furthermore, insects' visual systems respond over an extremely wide dynamic range of intensity changes encountered from night to day. Both behavioural and physiological experiments indicate that light adaptation of the insect's visual systems is performed at a number of levels from optics to neurons (Laughlin 1989; Laughlin & Hardie 1978). The earliest level where the light adaptability first occurs is the photoreceptors which have a sigmoidal stimulus response function referred to as a logarithmic compression (Warrant & McIntyre 1992). Consequently, the insect visual system can cope with a wide range of light intensity without saturating. This characteristic has already been mimicked into silicon photoreceptor circuits which have roughly five-order compression (i.e., log(output) = log(input) ÷ 5) (Tanner & Mead 1984; Delbrück & Mead 1991; Moini et al. 1993; Nguyen et al. 1993a).

2.4.3 Principle of Parallel Analog Processing

Insect visual systems, like other biological visual systems, have a highly parallel structure which is dominated by a retinotopic organization. It comprises three visual ganglia. Each neuropil is strictly organised into columns and strata. This structure offers a cluster of parallel processing channels which transmit analog signals from photoreceptors to the computation centre via synaptic operations (Laughlin 1987, 1984; Strausfeld 1989; Osorio &

Sobey 1992). In terms of hardwired neural circuits, this neural structure can be mapped into specialised analog computers, or filters, carrying out specific analyses of retinal signals in parallel. The key benefit of this architecture is to filter out unnecessary information and to process signals in parallel, thus providing sufficient communication bandwidth, and reducing the total amount of processing time. The effectiveness of this structure has been recently applied to design successful vision systems for mobile robots (Pichon et al. 1989; Nguyen et al. 1994a).

2.4.4 Principle of Visuomotor Control

Visual navigation of flies relies on neuronal motion detection. The visuomotor control process seems to be governed by a set of specific neurons that are sensitive to motion in particular directions (Hausen & Egelhaaf 1989). It has been shown that the fly *Drosophila* in a flight simulator can detect change from its straight flight course and quickly recover its stable flight using visual motion information (Wolf et al. 1992; Wolf & Heisenberg 1990). In insects, the control course is carried out by a dedicated crew of on-line image and flight-control processors consisting of about 100 million neurons. The overall principle is actually based on a coarse estimation of the optical flow field in the azimuthal plane (Franceschini et al. 1992).

In fact, the ability to detect visual motion is essential since most insects apply the principle of motion parallax to access the 3D structure of the environment (Horridge 1986; Lehrer et al. 1989; Srinivasan 1992a, 1992b; Sobel 1990a, 1990b). This principle seems to be well adapted to vision-guided mobile robots which are designed to move relatively freely in an unknown environment (Nguyen et al. 1994a; Abbott et al. 1994; Franceschini et al. 1992)

2.5 Summary

In this chapter we have reviewed evolutionary perspectives in visual motion computation. We briefly introduced Marr's hypothesis on vision as a process of extracting the 3D structure of the world. This purpose of vision led the field of visual motion to the paradigm *motion for structure* which has been predominant over the last decade, and has resulted in numerous algorithms on optical flow computation being proposed. These techniques, however, suffer from two primary problems, computational bottleneck and robustness, which make the existing theory on visual motion computation far from being useful for real time applications.

We also discussed *task-oriented vision* as an alternative for solving the above mentioned shortcomings. Within this paradigm, a set of visual tasks such as manipulation, mobility, navigation and recognition, among others, are attempted to be solved directly, using

the appropriate information, representation, and algorithms, along with the necessary hardware for each of these. This approach is strongly supported by evidence found in studies of biological visual systems, such as insect vision.

In fact, insect vision offers a number of lessons to machine vision including task-oriented vision, simple circuitry and automatic gain control, principle of parallel processing, and principle of visuomotor control. Some of these lessons are applied in designing the smart VLSI micro-sensors (see Chapters 7 & 8).

Chapter 3

The Insect Visual System and Motion Detection

3.1 Introduction

Motion detection is one of the most important functions of biological visual systems. Visual motion information serves many purposes such as the perception of depth using motion parallax, object discrimination or segregation, and navigational control. Motion cues are vital to fast moving animals like flying insects, whose visual systems are simple but efficient mechanisms for detecting and utilizing motion information. For example, the honey bee uses motion cues to explore the visual world and to estimate the distances separating it from nearby objects (Lehrer et al. 1988; Srinivasan 1992a, 1992b). Almost all insects react to optical stimulations presented to them by steering their eye, their head, or even their whole body (Horridge 1986). These reactions are termed *optomotor responses,* and serve to stabilize the insect's orientation with respect to the visual environment, and to simplify the detection of moving objects. In insects, optomotor responses are based upon direction-selective motion detection (Buchner 1976; Kirschfield 1972; Reichardt 1961, 1987).

It is not surprising, then, to discover motion-detecting units in the peripheral visual systems of animals. In the rabbit's retina, for example, there exist ganglion cells that are highly sensitive to the direction of motion (Barlow & Levick 1965). The invertebrate's visual system, for instance, has motion sensitive neurons which are giant tangential cells of the lobula plate in the fly visual system (Hausen 1981; Franceschini et al. 1989). These findings in biological visual systems have helped researchers to formulate a theory of motion detection which has led to a number of mathematical and practical models being proposed (Reichardt 1961; Barlow & Levick 1965; Bouzerdoum & Pinter 1989; Horridge 1990; Bouzerdoum 1993). These models represent a significant contribution to the understanding of the characteristics of the motion detection mechanism in biological visual systems.

In this chapter we give a brief overview of the insect visual system and of principles of visual motion detection in Section 3.2 and Section 3.3, respectively; for a more detailed review see Bouzerdoum & Pinter (1992). These principles are presented and combined into the form of the elementary motion detector. To illustrate these principles, two early neural models for motion detection derived from studies of insect vision are described in

Section 3.4, starting with the correlation motion detection model and its variants, followed by the shunting lateral inhibition model. Another model, namely the template model proposed by Horridge (1990), will be discussed in detail in the next chapter. The common aliasing problem associated with these models is also discussed. Finally, the chapter is concluded with a summary.

3.2 The Insect Visual System

The visual system of insects primarily consists of a pair of multifaceted eyes, referred to as compound eyes, each composed of approximately 3000 to 4000 ommatidia. Each ommatidium is a functional unit comprising a dioptric apparatus and a retinula, which accommodates eight photoreceptors labelled from R1 to R8. The insect visual system is anatomically constructed as a retinotopic organization which preserves the neighbourhood relationships when optical signals are projected from the retina to the layered structure of optic lobes. Each optic lobe essentially comprises three visual ganglia, namely the *lamina*, the *medulla*, and the *lobula complex*, which further divides into two synaptic layers: the *lobula* and the *lobula plate* as shown in Fig. 3.1.

3.2.1 The Lamina

The lamina is a curved plate lying beneath the retina, and is composed of structurally identical columnar subunits known as optic cartridges, which exactly match in number the overlying ommatidia. Each cartridge receives inputs from the group of eight photoreceptors (R1-R8) that share the same visual axis as the overlaying ommatidium, and projects outputs to the medulla column lying directly beneath it. For a detailed description of the lamina synaptic pathways, see Meinertzhagen & O'Neil (1991), Strausfeld (1989), and Shaw (1989). A cartridge is made up of five monopolar cells, L1 to L5, all of which extend axons to the next neuropil or ganglion, the medulla. The monopolar cells do not, however, form a homogeneous population because they differ in size, shape, and synaptic contacts. The cells L1 and L2 are the largest relay neurons in the lamina, and hence are named the *large monopolar cells* (LMCs). L5 is the smallest with minute fibres, and hence is called the *midget cell*. The LMCs are very similar in shape, and form the central elements of each cartridge.

The lamina contains no motion detection neurons, and its major output cells are concerned with coding contrast in order to assist the visual system in coping with large variations in background intensities. The responses of the LMCs, which are thought to provide inputs to the motion detection neurons of the second ganglion, i.e., the medulla, have the characteristics of either a low-pass or band-pass filter (Srinivasan et al. 1990). In particular, at high levels of ambient light the LMC produces a strongly biphasic response, consisting of

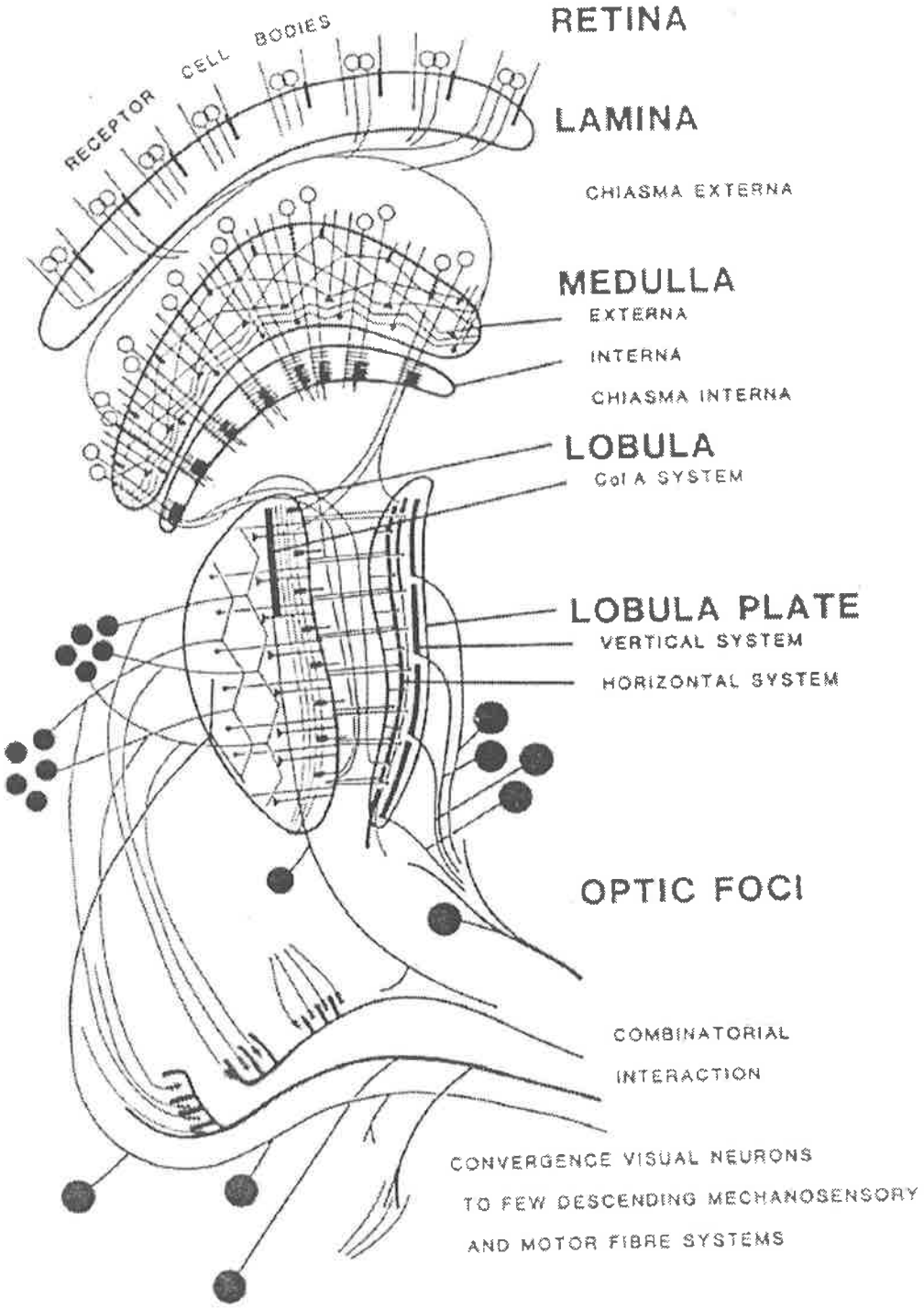


Fig. 3.1 The neuronal structure of the fly visual system's optic lobes (Adapted from Järvilehto 1985).

an initial hyperpolarizing phase followed by a depolarizing phase. The characteristics of this response resemble those of a band-pass filter. At low light conditions, however, the LMC yields a transient hyperpolarization in response to a brief flash of light delivered from a point source located anywhere within the receptive field. The characteristics of this response resemble those of a low-pass filter (Laughlin & Hardie 1978; Laughlin et al. 1987).

3.2.2 The Medulla

The second visual ganglion of the optic lobe, the medulla, also constitutes a retinotopically columnar structure, in which each column corresponds to a cartridge in the lamina. The neuronal synaptology, however, is more complicated than that of the lamina. The principle characteristics of the medulla are the large number and complex shapes of its neuronal cell types, which can be categorised into two general classes: the tangential cells and the perpendicular (columnar) neurons (Strausfeld 1976; Bouzerdoum & Pinter 1992). The tangential cells primarily contribute to an extensive network of lateral connections, whereas the perpendicular neurons serve to link the first to the third optic ganglia (Strausfeld 1976, 1989; Baussenwein et al. 1992).

Although little is known about the synaptology of the medulla, it is believed that the medulla is the insect visual system's most peripheral structure in which movement detection takes place (Devoe 1980). Motion detection is apparently carried out by two bushy perpendicular cells, named T4 and T5, which have axons extending to the wide-field motion-sensitive neurons (Strausfeld 1984). The medulla is anatomically the most complex structured ganglion of the optic lobe. This complexity is revealed by the variety of functionally different cells encountered in the region, ranging from simple contrast detectors to non-directionally and directionally motion-sensitive neurons (Bouzerdoum & Pinter 1992).

3.2.3 The Lobula Complex

The third optic ganglion in the fly, the lobula complex, is anatomically split into an anterior part, the lobula, and a posterior part, the lobula plate. This ganglion also possesses a columnar structure, although its retinotopic periodicity is coarser than that of the aforementioned ganglia. Each lobula column receives inputs from approximately six medullary columns. The lobula and lobula plate face each other.

A. The Lobula

The lobula receives inputs from medullary cells which also project into the lobula plate. The output neurons of the lobula convey signals to higher levels of the nervous system. The lobula is composed of several classes of regularly spaced small-field columnar neurons. At present, the functional characteristics of this part are still mysterious. However, from the structural features of the lobula, it is believed that these neurons comprise an array

of small-field units and are involved in the so-called *jump-* and *flight-initiation* circuitry (Hausen 1984).

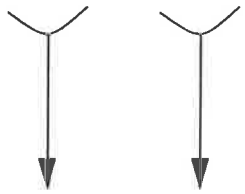
B. The Lobula Plate

The lobula plate is the motion computation centre of the fly visual system. It comprises anatomically two kinds of neurons: anaxonal amacrine cells and axonal tangential cells. The tangential cells constitute the principal anatomical characteristic of the lobula plate, and are primarily characterised by wide-field directionally-selective motion-detecting (DSMD) neurons (Hausen 1984; Strausfeld 1989). These DSMD neurons seem to share a common network of presynaptic elements derived from the medulla, and respond mainly to whole-field horizontal or vertical motion (Franceschini et al. 1989). Experimental evidence shows that the wide-field DSMD neurons of the lobula plate combine signals from large retinotopic arrays of small-field elementary motion detectors.

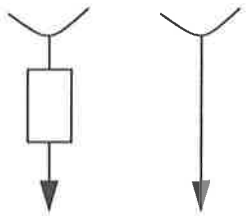
3.3 Principles of Visual Motion Detection: Elementary Motion Detector

Both behavioural and physiological experiments on insect visual systems indicate that movement detection occurs between neighbouring points of the sampling lattice of the eye (Kirschfield 1972; Franceschini 1989; Laughlin 1989; Laughlin et al. 1987; Baussenwein et al. 1992). This implies that a motion detector has two input channels, and the two channels should possess an asymmetric arrangement. Asymmetry is necessary for the detector to acquire direction selectivity. If the detector was symmetric, its input channels could then be interchanged without modifying its output signal. Without asymmetric preprocessing the detector would respond in the same way to motion in opposite directions; that is, it would lose its ability to respond differentially to motion in opposite directions. Furthermore, the interaction between the two channels must be nonlinear if the detector is to respond selectively to moving gratings, as the time-averaged output of a linear interaction is identical to the result of the interaction between the time-averaged inputs. The time-average input signals do not preserve the information concerning the temporal order of the incoming stimuli, and hence directional motion information is lost. Thus, a movement detector with a linear interaction cannot be directionally selective. Fig. 3.2 summarises three general requirements for a directionally sensitive motion detector. The asymmetrically nonlinear interaction between two input channels is termed the *elementary motion detector* (EMD) (Buchner 1976). Such an EMD will elicit a strong response when a visual stimulus moves in a specific direction (the *preferred* direction) and a weak response (or none at all) when the stimulus moves in the opposite direction (the *null* direction). The elementary motion detector is the mini-

Two inputs are necessary since motion is a vector which is represented by at least two points. A single detecting channel cannot discriminate a moving stimulus crossing its receptive field from left or right.



Asymmetry is necessary for the detector to acquire direction selectivity. If the detector were symmetric, the two input channels could then be interchanged without modifying the output. Accordingly, the detector would not be direction selective.



Nonlinear Interaction The interaction between the two channels must be nonlinear. Otherwise, the time-averaged output of a linear interaction is identical to the result of the interaction between the time-averaged inputs. The time-average input signals do not preserve the information about the temporal order of the incoming stimuli, and therefore the information about the direction of motion is lost.

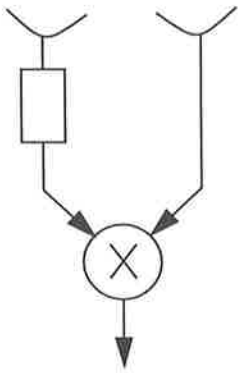


Fig. 3.2 A summary of general requirements for a directionally sensitive motion detector.

mum prerequisite for directionally selective detection of motion in the visual system. For more details, see (Poggio & Reichardt 1973; Egelhaaf et al. 1989; Borst & Egelhaaf 1989; Buchner 1976, 1984). These requirements are met by a variety of similar motion detection models (Torre & Poggio 1978; Marr & Ullman 1981; Reichardt 1961; Barlow & Levick 1965; Horridge 1990; Bouzerdoum 1991).

There exist primarily two general schemes for realizing the asymmetrically nonlinear interaction. The first is a veto operation proposed by Barlow & Levick in 1965, who discovered that inhibition is the mechanism underlying directional selectivity of ganglion cells in the rabbit's retina. The second scheme consists of activating the preferred direction by detecting a specific conjunction of excitation, and is illustrated in the correlation model derived by Hassenstein and Reichardt in 1956 (Bouzerdoum & Pinter 1992).

The initial stages of movement detection in insects appear to be based upon sequence discrimination by EMDs (Buchner 1976). Furthermore, overwhelming evidence has been accumulated showing that the DSMD neurons of the lobula plate receive both excitatory and inhibitory inputs from large retinotopic arrays of small-field EMDs which possess opposite preferred directions (Hausen 1984).

3.4 Neural Models for Early Visual Motion Detection

3.4.1 Correlation Model

The earliest and best-known motion detection model is a mathematical model proposed by Hassenstein and Reichardt in 1956, which accounts for the experimental data on the optomotor responses of insects (Bouzerdoum & Pinter 1991). In its simplest form, the primary operations of such a detector are summarised in Fig. 3.3. The nonlinear interaction is assumed to be a multiplication, M, which is the lowest possible nonlinear interaction (Borst & Egelhaaf 1989). The asymmetry is implemented by using a delayed unit, τ, and the input channels are spatially separated by the *sampling base*, $\Delta\varphi$, of the detector. When a moving stimulus first activates one input channel whose signal is delayed by an appropriate time, τ, the motion-induced signals in both channels tend to match at the multiplication stage, M, producing a strong response as shown in Fig. 3.3a. Conversely, when the moving stimulus fires the two channels in a temporally reversed order (corresponding to motion in the opposite direction) the distance between the motion-induced signals is further extended by the detector delay unit, and hence eliciting only a weak output (Fig. 3.3b). It is apparent that the optimal velocity of the detector, and consequently its dynamic range, depend strongly upon the time delay, τ, and the spatial sampling interval, $\Delta\varphi$. The smaller the time delay, the larger the optimal velocity and vice versa. Ultimately, this model is primarily construct-

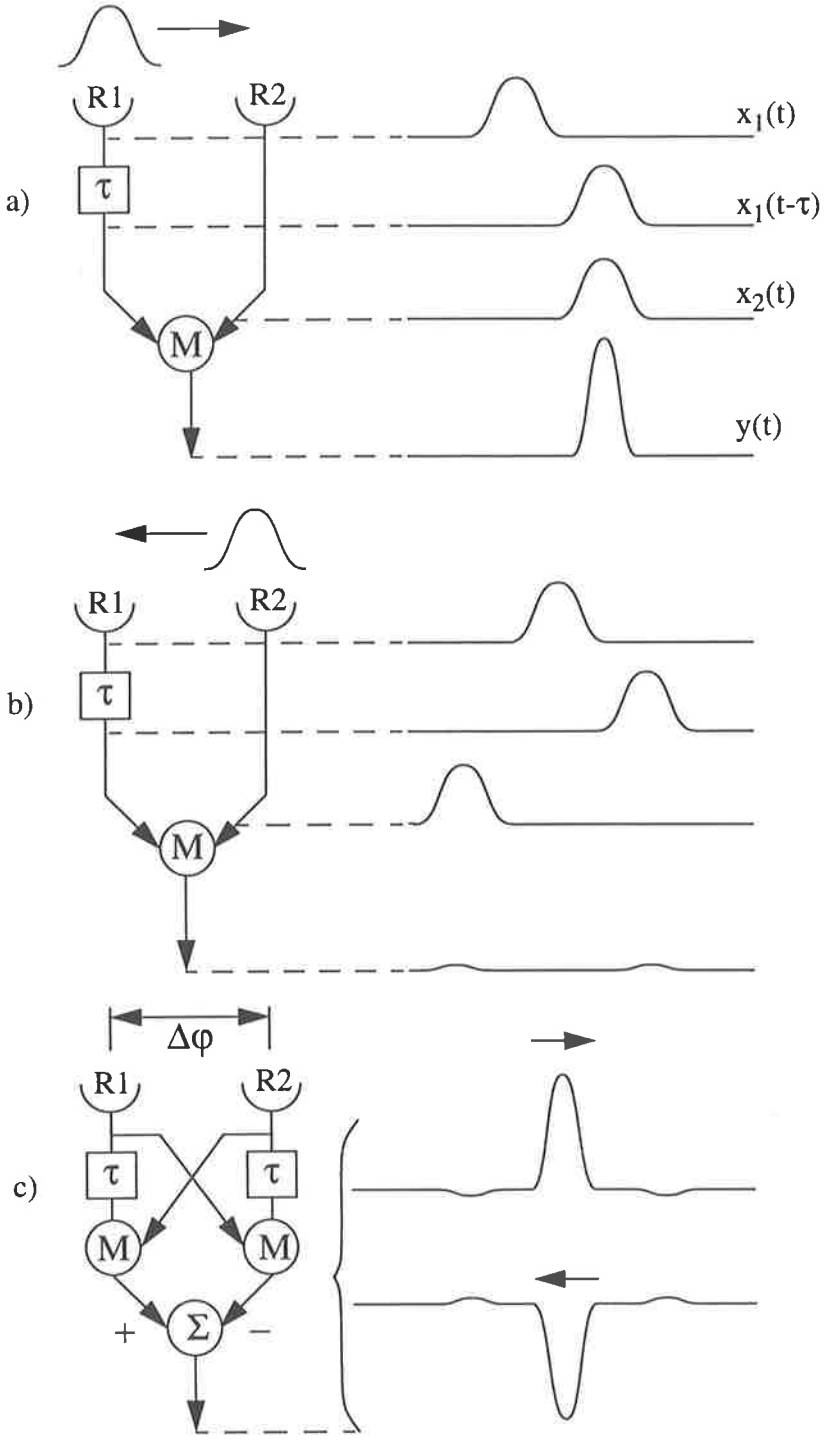


Fig. 3.3 Schematic representation of the correlation model. Δφ is the sampling base of the detector. R1 and R2 denote two adjacent photoreceptors, the τ-box represents a temporal delay unit, and the M-unit is the multiplication stage. a) Response of the subunit to a movement in a preferred direction. b) Response in the null direction. c) Responses of the full unit to a movement in either directions.

ed by the combination of a temporal delay and by the multiplication of the detector input channels. Alternatively, this type of detector could be described as measuring the degree of coincidence of the signals of its input channels, or performing a kind of *spatio-temporal cross-correlation.*

The motion detector shown in Fig. 3.3a and Fig. 3.3b are also sensitive to input signals which do not represent motion, such as background luminance. That is, the detector produces significant responses to input signals which are not induced by motion. To overcome this problem, a correlation-based movement detector may be composed of two mirror-symmetric subunits, each having a separate delay unit and multiplication operator. The outputs of both subunits are then subtracted as shown in Fig. 3.3.c, resulting in responses of the same amplitude but of opposite signs for motion in different directions. However, this is only correct if the detector is mathematically ideal, which is an unlikely assumption given the properties of the neuronal hardware. Thus, the correlation-based detector may not be sensitive to motion only, but can be expected to respond, at least to some extent, to temporal brightness modulations of a stationary stimulus ('flicker stimulation') (Borst & Egelhaaf 1989).

The output of this detector can be mathematically expressed by the equation

$$y(t) = x_1(t)\,x_2(t-\tau) - x_2(t)\,x_1(t-\tau) \tag{3.1}$$

where $y(t)$ is the output of the detector, $x_1(t)$ and $x_2(t)$ are the two inputs of the detector at time t.

Furthermore, the correlation-based detector might respond selectively not only to continuous motion, but also to non-continuous motion such as discrete jumps of the stimulus between the two input channels (Ullman 1981; Marr & Ullman 1981).

3.4.2 Variants of the Correlation Model

The original correlation detector was equipped with a linear time-invariant low-pass filter in each leg of the EMD as a delay unit. As a linear time-invariant filter delays different modulating input signals by different amounts of time, the Hassenstein-Reichardt detector employs two linear filters with short and long time constants, producing a near constant phase asymmetry between the channels over a practical range of temporal frequencies. Other researchers proposed to use a high-pass filter in one channel and a low-pass in the other in order to create the same phase lag between the channels over an extended range of modulation rates (Buchner 1976; Zaagman et al. 1978). A variety of correlation-based detectors have been formulated in different contexts by Thorson (1966), Michael (1968), Foster (1971), and van Santen & Sperling (1984).

The comparison process between signals in two adjacent channels is essentially fulfilled by multiplication followed by time integration, that is, first-order correlation. In fact, multiplication is the lowest nonlinear interaction mediating motion detection mechanism found in biological visual systems (Reichardt 1961; Buchner 1976, 1984; Zaagman et al. 1978). The multiplicative interaction of the correlation model received even more support from biological visual systems when Marmarelis & McCann (1973), investigating movement detection in class II neurons of the fly's visual ganglia, found that nonlinearities higher than second order do not contribute significantly to the detector response. At the same time, Poggio & Reichardt (1973) proved that if nonlinearities higher than second order are negligible, then any motion detecting system is equivalent to the correlation-based detector provided that infinite time integration of the output is employed.

3.4.3 Shunting Lateral Inhibition Model

Although the correlation model has been very widely used to investigate motion detection in insects, it still cannot explain certain phenomena found in insect visual systems such as non-saturation and adaptability to mean illuminance (Bouzerdoum & Pinter 1991). These features of the biological visual system are inherited from the role of shunting lateral inhibition at the single cell level. Another model, so-called *shunting lateral inhibition model*, which has recently been proposed, mimics the anatomical structure of the fly's visual system by operating with an asymmetric interaction mediated by the biophysical mechanism of

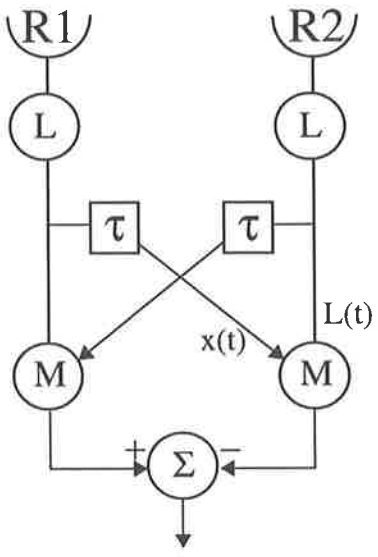


Fig. 3.4 Schematic representation of the shunting lateral inhibitory-based model. R1 and R2 are two adjacent photoreceptors, the L-units represent the lamina, and the M-units represent the medulla elementary movement detectors, the τ-units are temporal delay units.

shunting inhibition (Bouzerdoum 1991, 1993; Bouzerdoum & Pinter 1989, 1992, 1993).

Fig. 3.4 shows the basic functional structure of the shunting lateral inhibition model. The L-units are functionally similar to the large monopolar cells of the lamina, which produce transient responses as found in the insect visual system (Bouzerdoum 1993). The τ-units are temporal delay units and are implemented using low-pass filters (see, for example, Bouzerdoum 1991). Although the shunting lateral inhibition model is architecturally similar to the correlation model, shunting inhibition is the mechanism underlying the nonlinear interaction at the M-units, rather than a multiplicative operation. That is, the response of the M-units is governed by the differential equation

$$\frac{dm}{dt} = L(t) - am(t) - cx(t)\,m(t) \tag{3.2}$$

where a is a positive constant representing the passive decay rate of the M-units, $L(t)$ is the output of the L-units, $x(t)$ is the delayed signal carried by the lateral branch, and c is a positive constant representing a connection strength.

3.4.4 Spatial and Temporal Aliasing

The visual motion detection models discussed until now are susceptible to some form of aliasing. Since the correlation and shunting lateral inhibition models utilise the same principle for sampling the visual field, they are subject to the limitations on spatial resolution set by the Shannon sampling theorem. To demonstrate this problem, we only consider the response of the correlation-based model to a moving sine-wave. For the case of the shunting lateral inhibition-based model, details and discussion can be found in Bouzerdoum (1991). Let a drifting sine-wave grating be defined as

$$I(s,t) = I_o + \Delta I \cos(2\pi\mu f_s s - 2\pi f_t t + \varphi), \tag{3.3}$$

where I_o is the mean intensity, ΔI is the modulation amplitude, μ is the direction of movement (-1 for leftward motion, +1 for rightward motion), f_s and f_t are the grating spatial and temporal frequencies respectively, and φ is the initial phase. If $\Delta\varphi$ designates the difference in phase between two adjacent input channels, then it can be shown that the mean response of a correlation-based detector, $\overline{y(t)}$ †, is such that

$$\overline{y(t)} \propto \sin(\Delta\varphi)\sin(2\pi\mu f_s \Delta S) \tag{3.4}$$

†. The bar indicates that an infinite time average has been taken.

where ΔS represents spacing between the two channels. The sign of the mean response again depends upon the direction of movement, μ. It can be shown that if the spacing between the input channels ΔS is less than half the grating wavelength $\lambda\,(=\,1/f_s)$, (i.e. $\Delta S < \lambda/2$) then $\sin(2\pi\mu f_s \Delta S)$ has the same sign as μ. However, as the grating wavelength λ decreases $(\lambda/2 < \Delta S < \lambda)$, the sign of $\sin(2\pi\mu f_s \Delta S)$ changes, and the detector signals a wrong direction of motion. This problem is known as *spatial aliasing*. In summary, if the spatial sampling frequency, $1/\Delta S$, is smaller than the Nyquist spatial frequency $f_s/2$, then the detector suffers from the spatial aliasing problem according to the Shannon sampling theory.

The spatial aliasing phenomenon is also well known in the insect visual system. It was investigated and explained by Götz in 1964 using the concept of *geometrical interference* between the lattice of receptors, and the moving pattern causing a reversal of the direction of movement of the interference pattern by 180^{o} (Buchner 1984, 1976). However, the aliasing effect is dramatically alleviated since the sensitivity distribution of receptive fields of the photoreceptors in the insect visual system is a Gaussian. This distribution is a natural optical property of the eye, and acts as a spatial low-pass filter which removes all high frequencies in the input signals (Borst & Egelhaaf 1989). The concept of spatial low-pass filtering is also applied in designing visual motion detectors by adding spatial filters at the front of the detector input channels (van Santen & Sperling 1984, 1985).

Temporal aliasing, on the other hand, arises when $\sin(\Delta\varphi)$ reverses its sign, resulting in the wrong direction of movement being signalled. Applying the same technique for avoiding spatial aliasing, suitable temporal filters can be used to prevent the problem of temporal aliasing.

3.5 Summary

In this chapter we presented a brief overview of the anatomy and characteristics of the fly visual system, and described the principles of motion detection, which are the asymmetrically nonlinear interaction between two input channels. These principles should be satisfied by any model for detecting directional motion.

We also discussed the architecture of two different neural models for motion detection based upon the insect visual system. The correlation model is the earliest and was used to explain the characteristics of the optomotor responses of the fly. The shunting inhibition model was proposed to account for a number of features of insect visual systems. This model's dominant features are consistent with some of the phenomena found in biological visual systems, such as the adaptability to mean luminance without saturation.

However, both models have a common drawback, in that they all suffer some form of aliasing. Fortunately, the aliasing problem can be overcome by using the inherent property of optics, which is the aberration due to the diffraction of light waves when they enter optical devices such as lenses. This property creates the Gaussian-shaped receptive field of the photoreceptor, which may act like a spatial low-pass filter to remove high frequencies present in the input signals.

Chapter 4

Template Model and Its Directional Selectivity

4.1 Introduction

Significant progress has been made in the understanding of the insect visual system over the past 40 years. In the theory of motion detection particularly, different neural models provide different explanations to different phenomena in insect vision experienced by researchers (Chapter 3). Similarly, an empirical model, named *template model*, was recently proposed by Horridge (1990, 1991a, 1991b). The template concept developed directly from experimental studies of the behaviour and electrophysiology of the insect visual system, in particular from the rapid directional detection of sharp edges that jump by one photodetector distance (Horridge 1990).

Instead of performing a spatio-temporal correlation operation on signals between parallel channels as other well-known correlation models, such as the Reichardt model (Chapter 3), the template model detects changes in intensity and forms so-called *templates* by grouping intensity change responses of two adjacent receptors in two consecutive time steps. Responses to motion are interpreted by a number of special templates which were identified during the course of experiments. It was also claimed that the template model is related to the correlation model but there is no further analysis and no unique way for interpreting the physical meaning of template responses (Horridge 1990). In fact, the physical meaning of template responses to motion can be initially inferred from a proposed function, called *directional selectivity*, which provides a mathematical tool for the interpretation of templates. A heuristic investigation and the proposed formulation are together employed to identify template responses to real coherent motion, called *directionally motion-sensitive templates* (Nguyen et al. 1995).

In this chapter we describe the template model in Section 4.2. Section 4.3 and Section 4.4 discuss temporal and spatial aliasing problems. Section 4.5 quantifies the directional selectivity of this mode. Section 4.6 explains how templates sensitive to coherent motion are determined. Section 4.7 presents some experimental results of template responses to moving objects. Finally, the chapter ends with a discussion, where the features which make this model attractive to a hardware implementation are discussed.

4.2 Template Model

In essence, the visual field is spatially sampled into channels which are implemented as simple photodetectors. Knowing that the large monopolar cells of the lamina monitor changes in light intensity, the band-pass filters are employed to detect changes in optical input signals. The response of each filter is thresholded, sampled and quantized into one of three primitive states: intensity increase ($\uparrow$), decrease ($\downarrow$), and no change (–). The detection of directional motion in biological visual systems is carried out by a pair of adjacent receptors, and hence by copying this mechanism, the change states are spatially combined between adjacent channels. Thus, there are 9 possible spatial combinations: (– –), (–$\uparrow$), (–$\downarrow$), ($\uparrow$–), ($\uparrow\uparrow$), ($\uparrow\downarrow$), ($\downarrow$–), ($\downarrow\uparrow$), and ($\downarrow\downarrow$).

It has long been known that any visual system which detects directional motion must accomplish a spatio-temporal operation. In other words, it must perform a correlation in both time and space. This principle is applied in the template model where spatial combinations from two adjacent detecting channels are grouped at two consecutive time instances, t_1 and

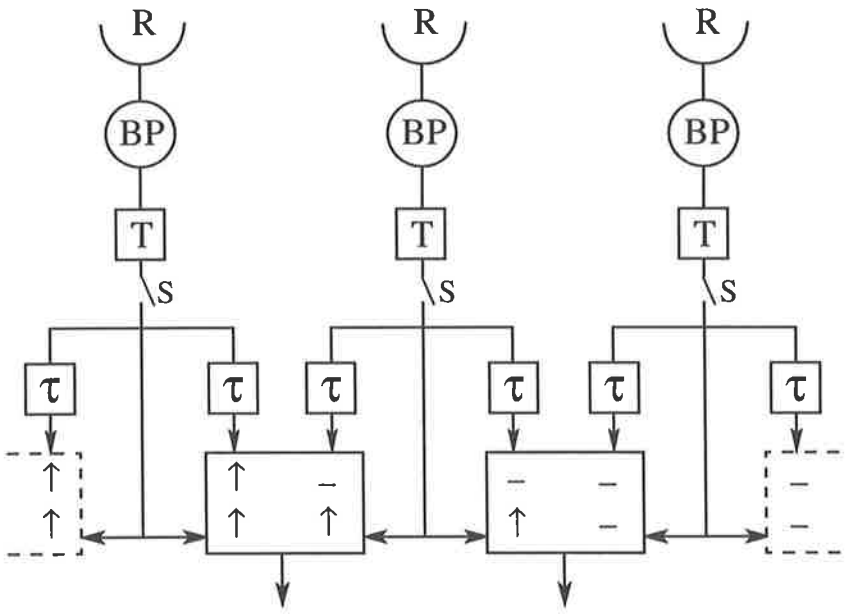


Fig. 4.1 Schematic representation of the template model. The R units represent the photoreceptors, the BP-units denote the bandpass filters to detect changes in intensity, the T-boxes represent the thresholding units. The S-switches are the sampling switches at a frequency of $1/\tau$ to quantize intensity changes into ($\uparrow$), ($\downarrow$), or (–), and the τ-boxes denote the temporal delay units with an amount of time τ.

t_2, yielding 81 possible spatio-temporal combinations, or *templates*. Fig. 4.1 depicts the template model.

In this model, motion detection is inferred from the meaning of the templates. Experimental evidence showed that only a small number of templates indicate directions of movements. The other templates do not signal directional motion. Fig. 4.2 illustrates some template responses.

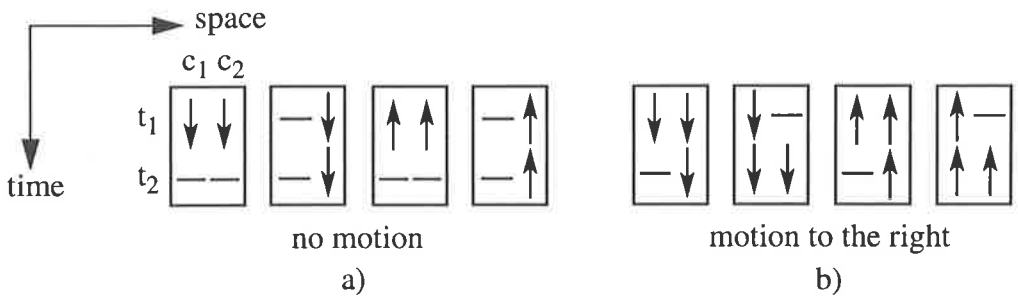


Fig. 4.2 Examples of template responses. a) Templates indicate no motion. b) Templates indicating motion to the right.

As the threshold operation is employed to further eliminate noise contamination in filtered signals, it might also remove small signals induced by slow motion or motion of low contrast objects, hence reducing the sensor's detectability. This is nothing but a trade-off. The higher the threshold level, the better the elimination of noise, but the lower the detectability of the sensor.

It is worth to note that the template model detects edges of moving objects rather than the objects themselves as moving edges cause changes in intensity. For a more detailed description, see Horridge (1990, 1991a, 1991b) and Sobey & Horridge (1990).

4.3 Temporal Aliasing

As discussed in Section 3.4.4 that the temporal aliasing problem suffered by the correlation model is caused by the phase difference between signals in adjacent channels. This temporal aliasing form seems to be not a problem in the template model since the primitive states of intensity changes are digitally delayed using latches in the template model. Thus, the template model avoids this temporal aliasing problem. However, the template model might suffer another form of the temporal aliasing due to the fact that signals of detecting channels are sampled. However, this temporal aliasing form can be avoided if the temporal sampling rate is kept at least twice of the bandwidth of the up-front bandpass filter.

4.4 Spatial Aliasing

Like the correlation model, the template model suffers the spatial aliasing problem if some kinds of spatial low-pass filtering are not applied. This problem is actually avoided in the physical implementations (Appendix A & Chapters 7 and 8) as the receptive fields of detecting channels have the form of the Gaussian low-pass filter and overlap with their two adjacent fields as shown in Fig. 4.3. The achievement of these receptive fields is discussed in details in Appendix A.

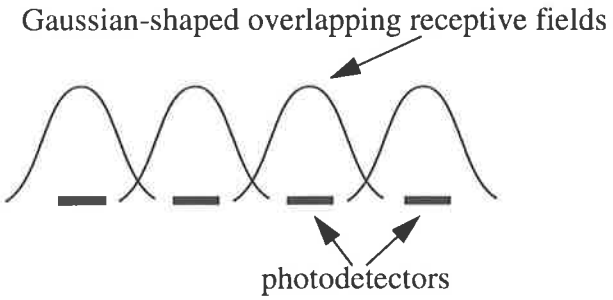


Fig. 4.3 Gaussian-shaped overlapping receptive fields of photodetectors.

4.5 Directional Selectivity

A visual motion detector is considered to be directionally selective if the detector is able to distinguish between movements in opposite directions. To have this ability, the detector should satisfy at least the three aforementioned conditions that are present in the EMD; that is, there exists an asymmetrically nonlinear interaction between two input channels. However, the satisfaction of all these conditions in the template model was not shown previously.

4.5.1 Early Interpretation of Directional Selectivity

Based upon experimental results, the directional selectivity of the template model was intuitively claimed as the indication of directions of motion exhibited by some special templates those possessing a 3:1 diagonally symmetric structure (Horridge 1991a, 1991b; Sobey & Horridge 1990). That is, templates have three identical entries of one primitive state and the other entry of another state as shown in Fig. 4.2.

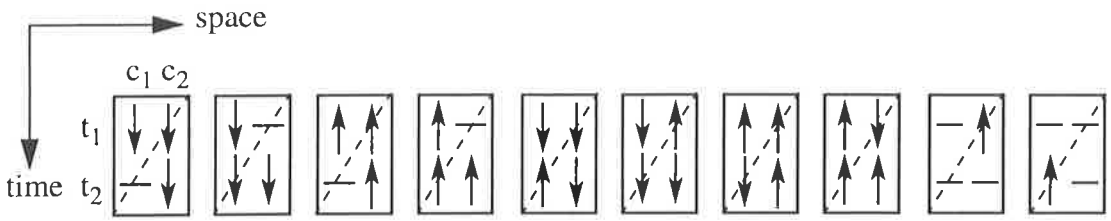


Fig. 4.4 Examples of template responses indicating motion to the right. These templates are 3:1 diagonally symmetric through the diagonal drawn from the top-right to bottom-left corners.

4.5.2 Observations

It is observed that:

1. There exists no explicit proof that the template model consists of a nonlinear interaction between any two input channels.
2. The lack of logical inference of the meaning of the templates led to confusion in the interpretation of template responses. There exist two different versions of the template interpretation; the former is in Sobey & Horridge (1990) and the later is in Horridge (1991a & 1991b). In Sobey & Horridge (1990), there is evidence that some 3:1 symmetric templates respond to motion in either direction such as the templates (–↑/↑↑) and (↑↑/–↑).
3. The architecture of the template model is similar to that of the correlation model except the explicit nonlinear interaction. That is, they all have the delay units to obtain the asymmetric arrangement between two adjacent channels.
4. From a mathematical point of view the template model works in the digital domain while the correlation model does in the analog domain.
5. An obvious difference between the two models is the presence of the bandpass filter in the template model. However, this difference is not a major issue because the correlation detector works even better if it is equipped with such bandpass filter. The main function of the filter is to eliminate the constant background illuminance which mainly results in the saturation phenomenon found in this detector (Egelhaaf & Borst 1989), and to prevent the temporal aliasing problem caused by high frequencies components. From the point of view of the biological visual system, the bandpass filter functionally works like the lamina, which makes the biological visual system sensitive to the contrast rather

than the absolute value of the illuminance. Furthermore, from a mathematical point of view, the presence of the bandpass filter does not dramatically alter the correlation detector's output (Egelhaaf & Borst 1989).

6. A major difference is the spatio-temporal nonlinear interaction between two adjacent channels. The correlation model employs a multiplication, while the template model results in spatio-temporal templates without an indication of such nonlinear interaction.

Based upon these observations it can be suggested that there may be a relationship between the two models. If there is then the template model is obviously qualified as a motion detection model.

4.5.3 Derivation of Directional Selectivity Function

The correlation model is now re-visited. However, it is now equipped with bandpass filters after the photodetectors to avoid the saturation phenomenon. The operation of the detector is still governed by Eq. (3.1), but the signals $x_1(t)$ and $x_2(t)$ are the outputs of the bandpass filters. If these signals are sampled with a frequency of $1/\tau$, Eq. (3.1) can be re-written in the discrete domain at time $t = n\tau$ as

$$y(n\tau) = x_1(n\tau)\,x_2(n\tau - \tau) - x_2(n\tau)\,x_1(n\tau - \tau) \tag{4.1}$$

where n is a positive integer. If τ is omitted for the sake of simplicity, Eq. (4.1) takes the form

$$y(n) = x_1(n)\,x_2(n-1) - x_2(n)\,x_1(n-1) \tag{4.2}$$

Eq. (4.2) simply indicates the response of the correlation detector at time $t = n\tau$. The response is either positive indicating movement in one direction, negative indicating movement in opposite direction, or null indicating no motion at all. Instead of obtaining an ideally null response when there is no motion, a weak output may be expected due to fluctuations in the background intensity or the imperfections of hardware, which are known as noise.

It is desirable to eliminate the weak output completely so that the sign of the output can be used directly to indicate directions of motion. For that purpose, a threshold function g may be employed as

$$Y_n = g(y(n)) \tag{4.3}$$

where $g(y(n))$ is defined in Eq. (4.4), where ΔY is a positive threshold level, and is graphically illustrated in Fig. 4.5. The sign of Y_n indicates directional motion directly.

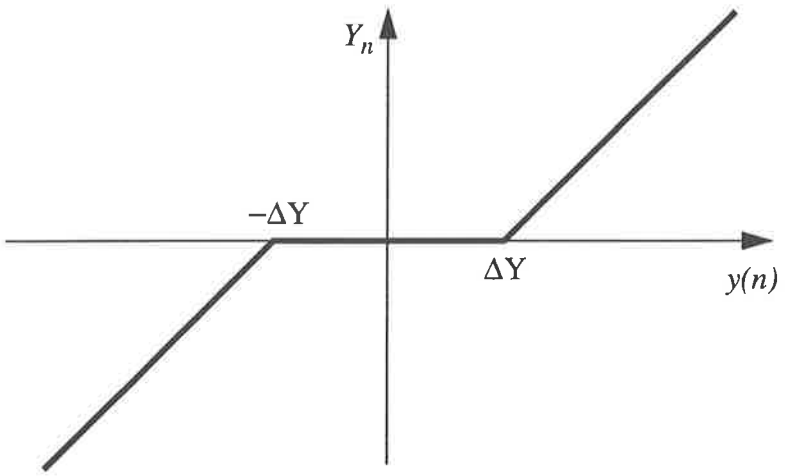


Fig. 4.5 Graphical representation of the proposed threshold function.

$$Y_n = \begin{cases} y(n) - \Delta Y & \text{if} \quad y(n) > \Delta Y \\ y(n) + \Delta Y & \text{if} \quad y(n) < -\Delta Y \\ 0 & \text{if} \quad |y(n)| \le \Delta Y \end{cases} \tag{4.4}$$

However, as the background fluctuations directly and strongly affect the detecting channels' responses (before multiplication) rather than the detector's final output (after multiplication), a similar threshold function, with a positive threshold level of ΔX, might also be applied to the detecting channels' responses such that

$$\begin{aligned} X_{1,2} &= \begin{cases} x_{1,2}(n) - \Delta X & \text{if} \quad x_{1,2}(n) > \Delta X \\ x_{1,2}(n) + \Delta X & \text{if} \quad x_{1,2}(n) < -\Delta X \\ 0 & \text{if} \quad |x_{1,2}(n)| \le \Delta X \end{cases} \\ X^d_{1,2} &= \begin{cases} x_{1,2}(n-1) - \Delta X & \text{if} \quad x_{1,2}(n-1) > \Delta X \\ x_{1,2}(n-1) + \Delta X & \text{if} \quad x_{1,2}(n-1) < -\Delta X \\ 0 & \text{if} \quad |x_{1,2}(n-1)| \le \Delta X \end{cases} \end{aligned} \tag{4.5}$$

By rearranging and substituting Eq. (4.5) into Eq. (4.2), Eq. (4.3) is obtained in the form of Eq. (4.6).

$$\begin{aligned} Y_n &= g\left(\left(X_1 \pm \Delta X\right)\left(X_2^d \pm \Delta X\right) - \left(X_2 \pm \Delta X\right)\left(X_1^d \pm \Delta X\right)\right) \\ &= g\left(\left(X_1 X_2^d - X_2 X_1^d\right) \pm \Delta X\left(\left(X_1 + X_2^d\right) - \left(X_2 + X_1^d\right)\right)\right) \end{aligned} \tag{4.6}$$

By comparing Eq. (4.6) with the definition of the threshold function in Eq. (4.4), it can be considered that

$$Y_n = X_1 X_2^d - X_2 X_1^d \tag{4.7}$$

and

$$\Delta Y = \Delta X\left(\left(X_1 + X_2^d\right) - \left(X_2 + X_1^d\right)\right) \tag{4.8}$$

In fact, Eq. (4.7) is a discrete form of the correlation model, Eq. (3.1), with a minor alteration where a threshold operation is applied to the responses of the detecting channels. Thus, the output of Eq. (4.7), which is either positive, negative, or null, still indicates directional motion at that sampling instant.

On the other hand, a template response, T, in the template model can be considered as a 2×2 matrix,

$$T = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \tag{4.9}$$

where each entry, a_{ij}, is assigned one of the following values:

$$a_{ij} = \begin{cases} 1 & \text{for increase} \\ 0 & \text{for no-change} \\ -1 & \text{for decrease} \end{cases} \tag{4.10}$$

Similar to the discrete form of the correlation model in Eq. (4.7), a directional selectivity function (DSF) f first reported by Nguyen et al. (1995), of the template model is proposed as follows:

$$f(T) = \begin{vmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{vmatrix} = a_{11}a_{22} - a_{12}a_{21} \tag{4.11}$$

This DSF, which maps templates into the set of integers, Z, actually calculates the determinant of a template matrix. The DSF is well defined and can only take the following values {-2, -1, 0, +1, +2}. Table 4.1 shows the values yielded by this function for all template matrices. Templates whose entries are either vertically or horizontally symmetric have a zero determinant. The table is symmetric about the main diagonal. The DSF divides tem-

plates into either positive, negative, or null responses which can be used to classify templates into three distinct groups corresponding to motion to the right, to the left, and no motion, respectively.

	– –	–↑	–↓	↑–	↑↑	↑↓	↓–	↓↑	↓↓
– –	0	0	0	0	0	0	0	0	0
–↑	0	0	0	1	1	1	-1	-1	-1
–↓	0	0	0	-1	-1	-1	1	1	1
↑–	0	-1	1	0	-1	1	0	-1	1
↑↑	0	-1	1	1	0	2	-1	-2	0
↑↓	0	-1	1	-1	-2	0	1	0	2
↓–	0	1	-1	0	1	-1	0	1	-1
↓↑	0	1	-1	1	2	0	-1	0	-2
↓↓	0	1	-1	-1	0	-2	1	2	0

Table 4.1 Determinant values of template matrices in the range of [-2,2].

Although the discrete form of the correlation model works on analog data (after thresholding) of the entries, while the template model works on digital data (quantized into one of three levels) of the entries, the sign of the responses of the two detectors, in principle at least, can be used for detecting directional motion according to the EMD principles. In other words, the template model does satisfy the requirements for a directional motion detection by evaluating the directional selectivity of templates using the defined directional selectivity formulation. Thus, the template detector differentially responds to directional motion.

4.6 Directionally Motion-Sensitive Templates

The template model is functionally similar to the correlation model and the directional selectivity is governed by the DSF, Eq. (4.11), which ensures, at least in principle, that any template whose corresponding matrix has a non-zero determinant is sensitive to directional motion. As a result of a preliminary determination based upon the DSF, there are 48 motion-sensitive templates according to the number of non-zero determinant template matrices. These motion-sensitive templates are further classified into two distinct sets corresponding to two possible directions of motion; each has 24 templates which have either the positive or the negative sign output. This means that there might be 24 different types of motion oc-

curring in the same direction. In other words, the output of the DSF only indicates a particular direction of motion, but does not distinguish the nature of the templates. For example, the positive sign response indicates motion in one direction but cannot demonstrate whether the motion is continuous or discretely jumped.

Fortunately, with the template model where templates are clearly pre-determined, it is possible to inspect all these 24 combinations, and then classify the nature of possible motion, that is, what are the conditions for these templates to be formed.

4.6.1 Templates Sensitive to Coherent Motion

Firstly, coherent motion means that signals detected by two adjacent channels must be induced by the same object, that is, motion-induced signals in both channels must have the same polarity such as either both increase or both decrease.

Secondly, it is desirable but very difficult to show that signals in adjacent channels are really induced by the same object. However, experimental evidence shows that if motion-induced signals in adjacent channels are overlapping they are certainly generated by continuous movement of one object rather than by any source as shown in Fig. 4.6. Experimental results, from either simulations (latter in this chapter) or real-time experiments (Chapters 7 & 8), show that templates exhibiting the overlap between the motion-induced signals in adjacent channels are robust templates which indicate coherent motion. The overlapping condition, in fact, ensures that motion-sensitive templates have the 3:1 diagonally symmetric structure.

In summary, templates which are sensitive to coherent motion must satisfy two criteria below.

1. Motion-induced signals in adjacent detecting channels must have the same polarity.

2. There should exist an overlap between polarity-alike signals in adjacent channels, that is, those templates possess the 3:1 diagonally symmetric structure.

Furthermore, there may be different kinds of background such as that with either stable intensity or varying intensity. In this study, the background is assumed to have stable intensity. The reason for this assumption will be clarified later in this section. This assumption simply alters the claim on the structure of the 3:1 diagonal symmetric motion-sensitive templates from a general statement (Section 4.5.1) to a more specific statement that *templates sensitive to coherent motion are the ones in which one of the four entries is no-change in intensity and the other three must be identically either increase or decrease in intensity.* These templates are named as directionally motion-sensitive templates (DMSTs).

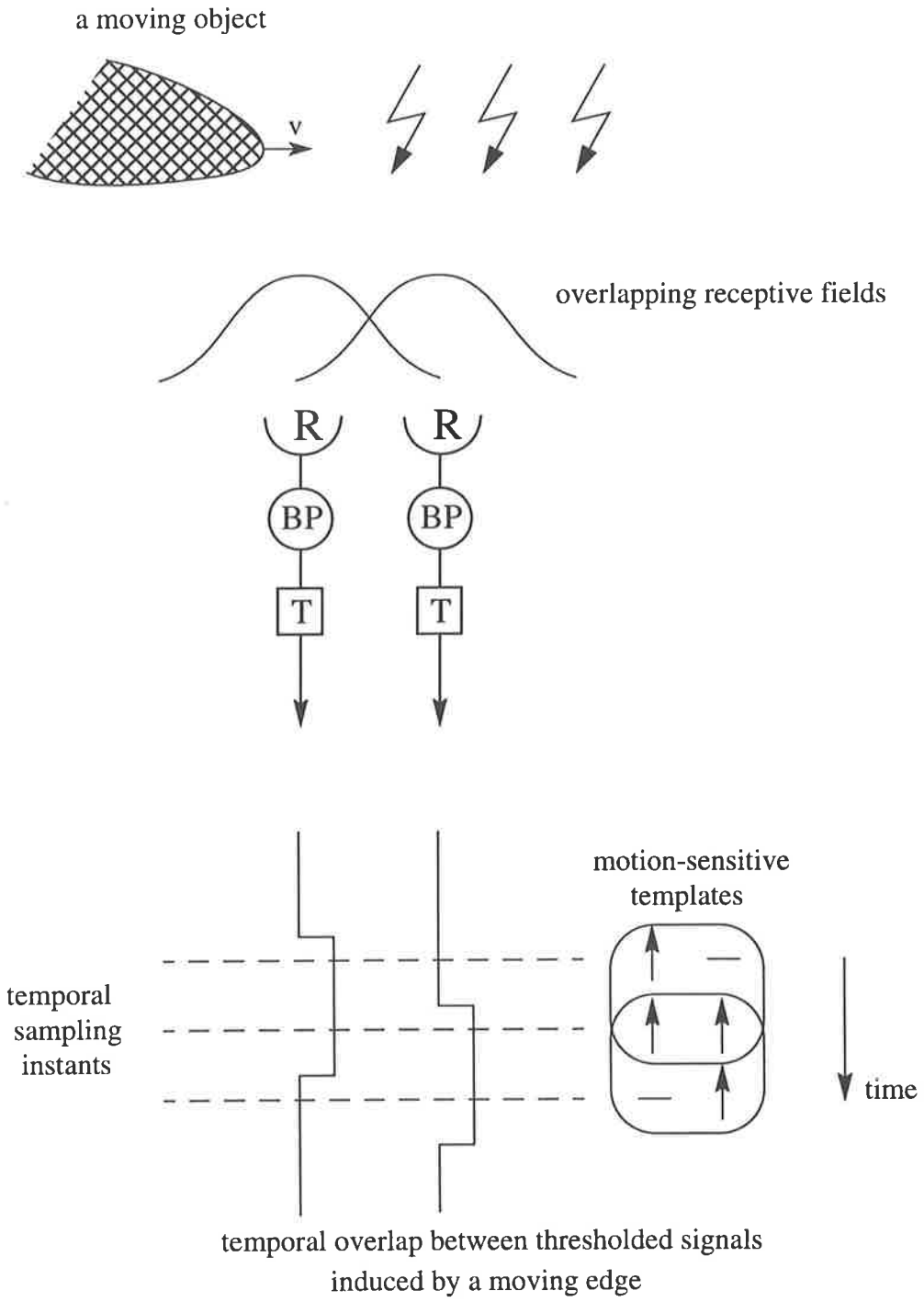


Fig. 4.6 The overlap between individual receptive fields can ensure that motion-induced signals between detecting channels are also overlapped.

4.6.2 DMST Characterisation

In the matrix form †, the DMSTs are in the forms below.

$$\begin{bmatrix} 0 & \pm 1 \\ \pm 1 & \pm 1 \end{bmatrix}, \begin{bmatrix} \pm 1 & 0 \\ \pm 1 & \pm 1 \end{bmatrix}, \begin{bmatrix} \pm 1 & \pm 1 \\ \pm 1 & 0 \end{bmatrix}, \begin{bmatrix} \pm 1 & \pm 1 \\ 0 & \pm 1 \end{bmatrix}$$

In the template model, the eight DMSTs are further characterised into four groups according to four combinations of the edge polarity (dark-to-bright or bright-to-dark ††) and motion direction (to the left or to the right). Thus, each group has two DMSTs. For example, the following two templates are in the group of a bright edge with respect to a dark background moving to the right.

$$\begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}, \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}$$

In the context of moving objects, consider a bright object with respect to its background moving across a detecting channel from left to right. The response of the channel's photodetector is rising at the *onset* of the object, and is falling at the *offset* of the object. If the size of moving objects are sufficiently large so that they can cover the receptive field of the detecting channel, the onset corresponds to the enter of the objects' leading physical edge, while the offset corresponds to the leaving of the object's trailing physical edge. Subsequently, two differential pulses are generated by the channel's bandpass filter as shown in Fig. 4.7. The first template in the example above occurs at the differentially outer edge of the first differential pulse, while the second template occurs at the differentially inner edge of the same pulse. Thus, the second template can be considered as a redundant one of the first template. However, the second template does occur if there is a sufficient separation between the two differential pulses, that is, there exist some intensity no-changes between the two in order to ensure one no-change entry present in this template. In other words, the object should take a sufficient amount of time to pass through one detecting channel so that the two differential pulses are separated. This means that the appearance of the second template depends on the speed of the object.

The concept of the differentially inner and outer edges of the differential pulses is em-

---

†. Each template matrix represents two possible templates by taking the signs of the entries from either top or bottom level but not cross each other. For example, $\begin{bmatrix} 0 & \pm 1 \\ \pm 1 & \pm 1 \end{bmatrix}$ represents for two templates: $\begin{bmatrix} 0 & 1 \\ 1 & 1 \end{bmatrix}$ and $\begin{bmatrix} 0 & -1 \\ -1 & -1 \end{bmatrix}$.

††. An bright-to-dark edge is conceptually defined as a dark side is followed a bright side in the direction of its motion and vice versa.

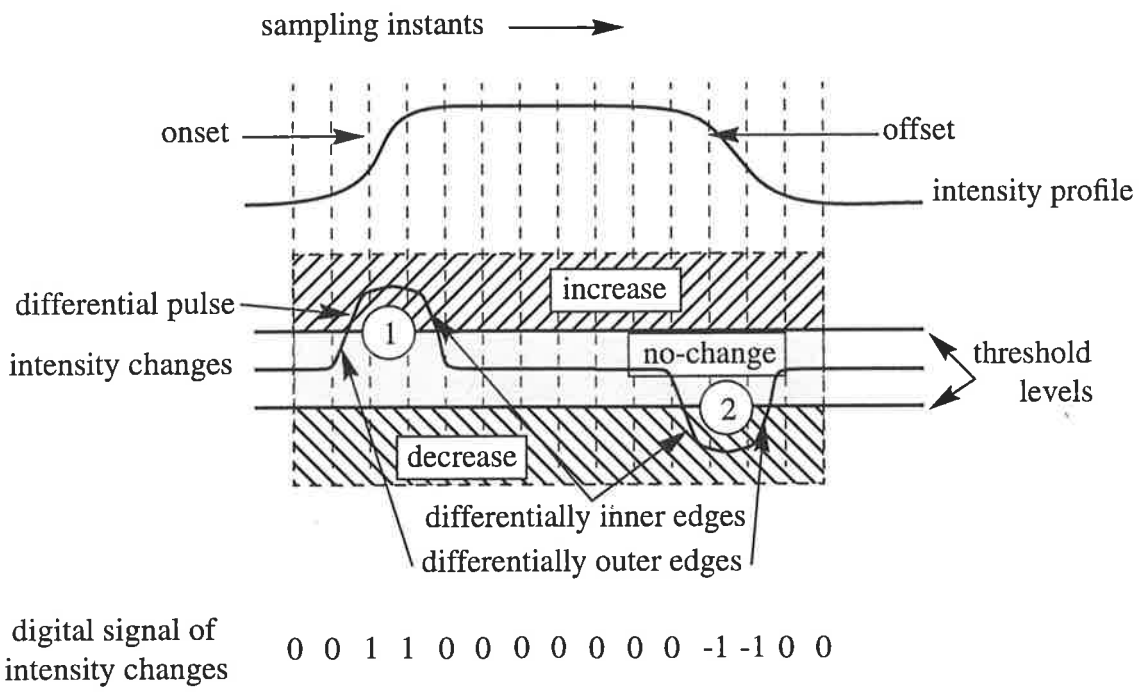


Fig. 4.7 The concept of differential pulses, 1 & 2, and the differentially inner and outer edges of the intensity-changed signal in one channel along the temporal axis.

ployed to clarify the positions where the DMSTs occur. If these differential pulses from adjacent detecting channels are spatially stacked together, two strips of intensity increases and decreases are formed. Consequently, these differentially inner and outer edges in the spatio-temporal domain form so-called *virtual inner* and *outer motion paths* respectively. Thus, there are four virtual motion paths for each moving object. The virtual outer paths always correspond to the onset and the offset of the object. The two virtual inner paths might not be recognised by the DMSTs if there is no sufficient separation between the two differential pulses.

In summary, there are only four pairs of directionally motion-sensitive templates which can be used reliably to indicate directional motion as shown in Fig. 4.8. Each pair indicates motion of one edge in one direction. These DMSTs are templates possessing the 3:1 diagonally symmetric arrangement in which there are one no-change entry and three identical entries, either increase or decrease.

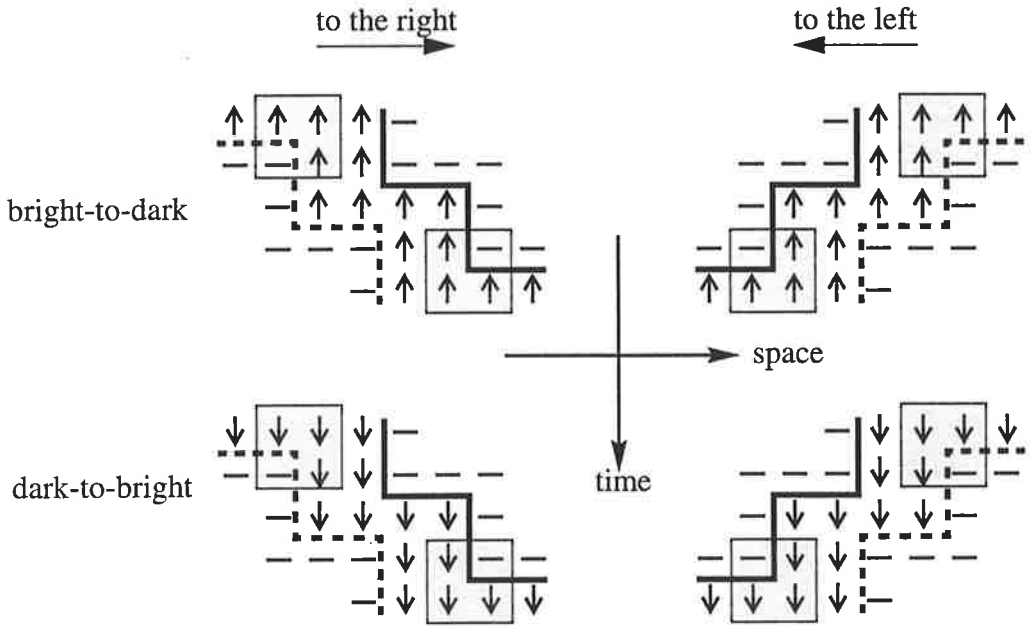


Fig. 4.8 Four groups of directionally motion-sensitive templates. The groups are four possible combinations formed between two motion directions (left or right) and two edge polarities (light-to-dark or dark-to-light). Solid staircases are virtual outer motion paths. Dashed staircases are virtual inner paths.

4.6.3 Other Templates Possessing a Non-Zero Determinant

Other templates mean those templates which are responses due to reasons other than coherent motion. Such reasons could be noise, discrete jumped motion, or fast motion and so on.

A. Different Polarities

Although the twelve template matrices have non-zero determinants, signals in two channels have different polarities. Hence, they cannot indicate the presence of coherent motion, and are possibly generated by movements of two different objects sensed by two adjacent detecting channels.

$$\begin{bmatrix} 0 & \pm 1 \\ \mp 1 & 0 \end{bmatrix}, \begin{bmatrix} \pm 1 & 0 \\ 0 & \mp 1 \end{bmatrix}, \begin{bmatrix} 0 & \pm 1 \\ \mp 1 & \pm 1 \end{bmatrix}, \begin{bmatrix} \pm 1 & 0 \\ \pm 1 & \mp 1 \end{bmatrix}, \begin{bmatrix} \pm 1 & \mp 1 \\ \pm 1 & 0 \end{bmatrix}, \begin{bmatrix} \mp 1 & \pm 1 \\ 0 & \pm 1 \end{bmatrix}$$

B. Non Overlap

The four following templates, in principle, can indicate directional motion since their DSFs have non-zero values. However, these templates may be obtained from discrete jumped motion, possibly caused by two separate objects, rather than by temporally continuous motion of one object as the signals are not overlapped. Experiments also showed that such templates may be responses due to flicker noise (Horridge 1990).

$$\begin{bmatrix} 0 & \pm 1 \\ \pm 1 & 0 \end{bmatrix}, \begin{bmatrix} \pm 1 & 0 \\ 0 & \pm 1 \end{bmatrix}$$

C. Fast Motion or Flicker Noise

The signals in two channels of the templates below do not have the same polarity. The non-zero signal in one of the channels is possibly a result of a fast motion or motion of small objects as illustrated in Fig. 4.9. However, signals generated by fast movements in general cannot be distinguished from flicker-noise (Horridge 1990). Thus, the uncertainty in identifying the source generating these templates makes them unreliable to be used as DMSTs.

$$\begin{bmatrix} 0 & \pm 1 \\ \pm 1 & \mp 1 \end{bmatrix}, \begin{bmatrix} \pm 1 & 0 \\ \mp 1 & \pm 1 \end{bmatrix}, \begin{bmatrix} \pm 1 & \pm 1 \\ \mp 1 & 0 \end{bmatrix}, \begin{bmatrix} \pm 1 & \pm 1 \\ 0 & \mp 1 \end{bmatrix}$$

$$\begin{bmatrix} 0 & \pm 1 \\ \mp 1 & \mp 1 \end{bmatrix}, \begin{bmatrix} \pm 1 & 0 \\ \mp 1 & \mp 1 \end{bmatrix}, \begin{bmatrix} \mp 1 & \mp 1 \\ \pm 1 & 0 \end{bmatrix}, \begin{bmatrix} \mp 1 & \mp 1 \\ 0 & \pm 1 \end{bmatrix}$$

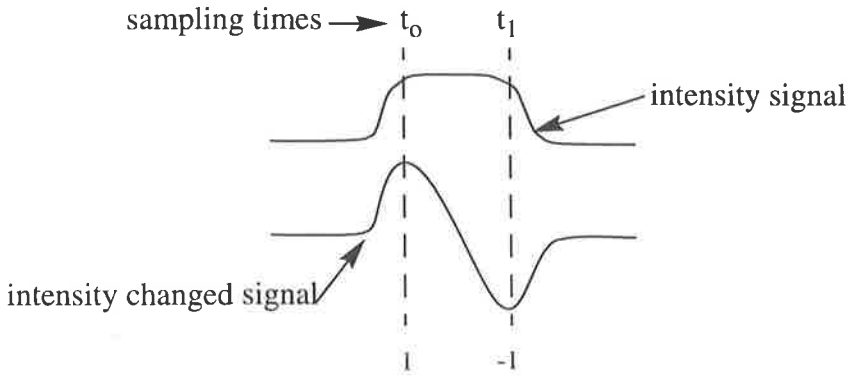


Fig. 4.9 Intensity signal, solid curve, is resulted from fast motion or motion of small objects. Differentiated signal, dotted curve, and digital signal, 1 and -1, are formed.

D. Background with Unstable Intensity

Finally, templates which have a determinant of either +2 or –2 take the following forms.

$$\begin{bmatrix} \mp 1 & \pm 1 \\ \pm 1 & \pm 1 \end{bmatrix}, \begin{bmatrix} \pm 1 & \mp 1 \\ \pm 1 & \pm 1 \end{bmatrix}, \begin{bmatrix} \pm 1 & \pm 1 \\ \pm 1 & \mp 1 \end{bmatrix}, \begin{bmatrix} \pm 1 & \pm 1 \\ \mp 1 & \pm 1 \end{bmatrix}$$

As all four entries of these templates indicate changes in intensity, these templates must be generated by motion. If the three polarity-alike entries are considered as being generated by one moving object, the fourth polarity-opposite entry must represent for the background. This entry however indicates that the background is of varying intensity which may be due to flicker-noise or object occlusions. The fourth entry will become no-change if the background becomes stable, and hence these templates become DMSTs. This means that those templates which have determinants of –2 or +2 may be of the forms of DMSTs with the background of varying intensity.

Nonetheless, if these templates are used for detecting directional motion, they lead to confusion in the interpretation of these templates on the edge polarity and position. Consider a bright object moving to the right with an intensity-decreased background as shown in Fig. 4.10 for instance. As a result, there are two different templates, 1 and 2, of this category. The template 1 indicates that motion of the bright object's leading edge as considered. In contrast, the template 2 may be used to indicate motion of a dark object's trailing edge moving to the right in an intensity-increased background.

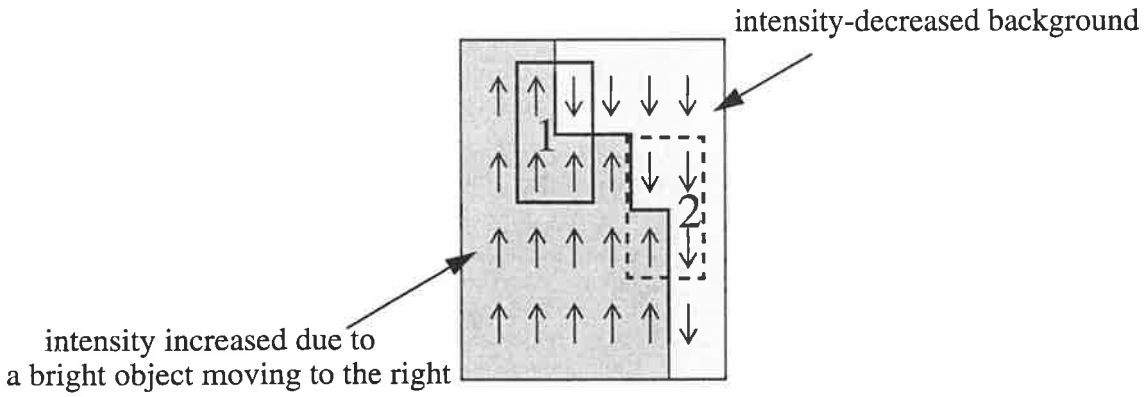


Fig. 4.10 Templates responses due to a bright object moving to the right with an intensity-decreased background.

Thus, this example shows that if the templates of this category are used to detect directional motion, they may result in a confusing interpretation of moving objects and their positions. This is the main reason for which the background with stable intensity only is considered in this study.

4.7 Response to Moving Objects

The template model has been extensively tested with software simulation and also with designed hardware sensors. The designs of the hardware sensors are described in more detail in the second part of this thesis, i.e., Chapters 7 and 8. The software simulation is discussed in Appendix B.

In order to enhance the display visualization, results, from either software simulation or real-time experiments, are displayed in two forms: changes in intensity and DMSTs. In the former form, intensity increases are denoted as 'i', intensity decreases are represented with 'd', and intensity no-changes are displayed with dots ('.'). In the later form, the eight DMSTs are displayed using the first eight alphabetic letters from 'A' to 'H' as summarised in Table 4.2. For other templates which do not represent motion, dots ('.') are also used to display.

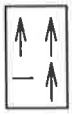
Polarity	Motion to the right		Motion to the left	
	Display code	Template	Display code	Template
bright-to-dark	A		E	
	B		F	
dark-to-bright	C		G	
	D		H	

Table 4.2 Alphabetic display codes of the DMSTs.

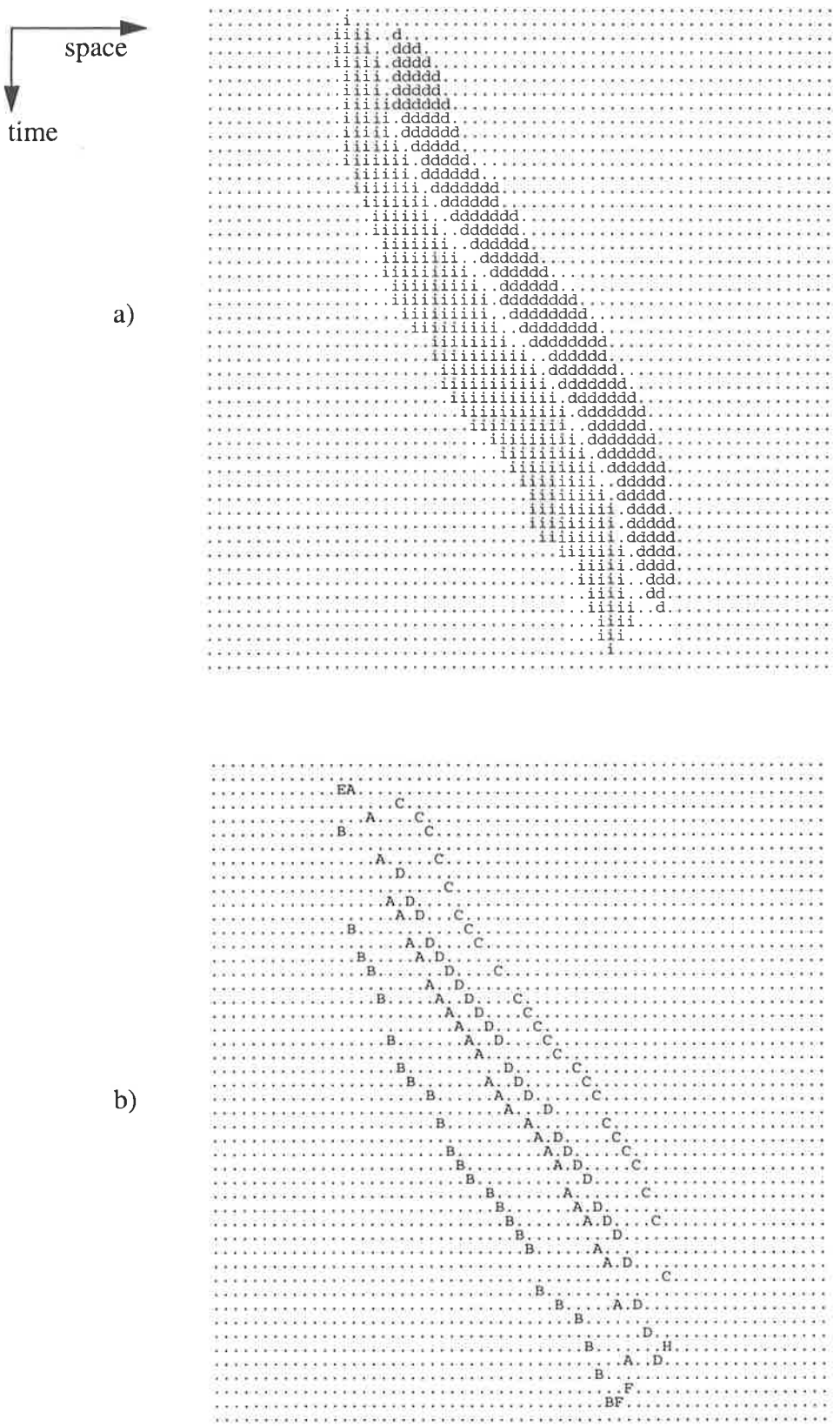


Fig. 4.11 Motion patterns of a hand moving to the right at a distance of 1m. a) Changes in intensity. b) Directionally motion-sensitive templates.

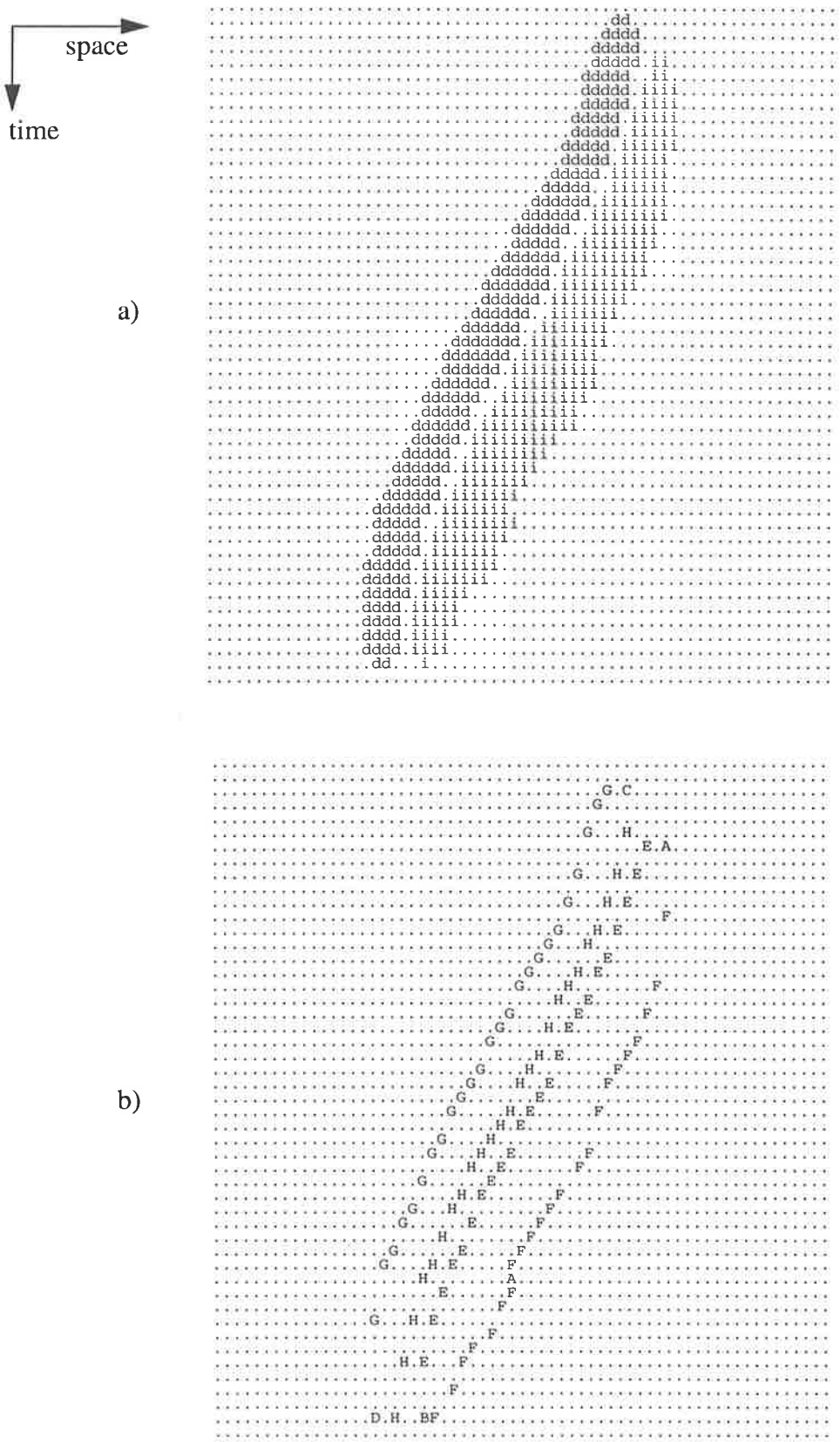


Fig. 4.12 Motion patterns of a hand moving to the left at a distance of 1m. a) Changes in intensity. b) Directionally motion-sensitive templates.

4.7.1 Examples

Following are the results obtained from real-time experiments of the sensor II (see Chapter 8) showing motion of a hand at a distance of 1 metre moving to the right in Fig. 4.11 and to the left in Fig. 4.12 in both forms of intensity changes and DMSTs only.

The real-time experimental results confirm the following:

- The onset and the offset of moving objects create two differential pulses which can be recognised as two strips of changes in intensity. One is increasing in intensity and the other is decreasing in intensity.
- These two strips create four virtual inner and outer paths which can be recognised by templates: 'A', 'B', 'C', 'D' for motion to the right, and 'E', 'F', 'G', 'H' for motion to the left.

Directional motion can be recognised by eight directionally motion-sensitive templates from 'A' to 'H'.

4.8 Discussion

In this chapter the template model is presented with a discussion on the temporal and spatial aliasing problems. It is recognised the fact that the template model was originally not shown to be satisfied the principles of visual motion detection, hence leading to a confusing interpretation of template responses. This problem is solved by the development of a mathematical formulation of the directional selectivity of the template model, which aids in showing that the template model is functionally similar to the correlation model. Using this formulation together with a heuristic investigation, eight DMSTs are identified with confirmations from the real-time experimental results.

The first advantage of the template model is the explicit interpretation of template responses, which makes it possible to identify responses due to coherent motion from other responses due to, for example, flicker-noise or discretely jumped motion. The second advantage is the hardware simplicity (Chapters 7 & 8). Obviously, a template detector does not require a multiplication unit to perform a correlation operation compared to a correlation detector. Circuits implementing the detection stage of changes in intensity do not require a high precision as they only qualitatively monitor any local changes in intensity. Furthermore, as the signals from the detection stage are digitized early, template processing can be carried out in the digital domain. This fact makes the processing stage of a detector more robust, hence increasing the overall robustness of a template detector.

Chapter 5

Feature Representation of Motion Trajectories

5.1 Introduction

Visual systems, either biological or machine, are usually overloaded with a vast amount of information received from photoreceptors. However, only a tiny fraction of this amount is really relevant to a given task, say motion detection for example. Thus, extracting information which is irrelevant to a given task is essential in the visual systems as stated by Dickmanns & Graefe (1988a) "*Limiting the number of features to be processed is not meant to exclude useful redundancy, which is absolutely necessary for any robust system, but rather to avoid wasting time or computing resources on processing irrelevant parts of a scene.*"

In nature, this kind of specificity of extracting features is intrinsically utilised in biological vision systems. For example, insect visual systems have special neurons dedicated to detecting directional motion (Franceschini et al. 1989; Hausen 1984). In other words, these neurons act like filters to remove information which are not induced by motion.

In machine vision, following the task-oriented vision approach (Chapter 2) there are some vision systems which have been built for specific tasks (Nguyen et al. 1994a; Yakovleff et al. 1993; Nelson & Aloimonos 1989; Pichon et al. 1989). Visual models to be implemented in these machine vision systems are carefully provided with abilities of extracting features which are necessary for a given task, hence achieving a reduction in data as early as possible within the models rather than in subsequent processing stages.

In visual motion detection systems particularly, the reduction in data can be achieved by representing motion trajectories of moving objects with a number of features rather than complete trajectories in a sequence of time-varying images (Nguyen et al. 1995). Using the template model (Chapter 4), a number of selected templates can be used as features for representing motion trajectories.

In this chapter we present two different schemes for feature representation of motion trajectories within the framework of the template model. To a small extent, the material in this chapter has been previously reported in Nguyen et al. (1993b & 1995). Section 5.2 begins with a brief review of the spatio-temporal approach used in motion detection.

Section 5.3 describes how motion is considered as orientation in the spatio-temporal domain. From the template model point of view, Section 5.4 illustrates the staircase shape of motion trajectories. Section 5.5 shows how two different schemes of the feature representation of motion trajectories, which are then confirmed by simulation and experimental results in Section 5.6 and Section 5.7 respectively. The chapter ends with a summary.

5.2 Spatio-Temporal Approach

Analysis in the spatio-temporal domain plays an important role in the study of visual motion. From an early stage, researchers in biological vision have performed motion analysis in the spatio-temporal domain to account for data from both psychophysical studies and single-cell electrophysiological recordings on the retina, lateral geniculate nucleus, and visual cortex. Barlow & Levick (1965) suggested that some neurons in the rabbit retina have directional selectivity which can be interpreted by a model, similar to the Reichardt's, although signals from two adjacent channels are subtracted instead of being multiplied. Emphasis on this analysis in the spatio-temporal domain is not surprising since these researchers were faced with the astonishing fact that biological visual systems are capable of representing the position of an object as an almost continuous function of time, even when the image itself is presented at discrete positions at separate instants (Barlow 1979; Anstis 1980; Ramachandran & Anstis 1986). Barlow (1979) reported that time and position can play interchangeable roles in determining the apparent position of a moving object in the cat's visual cortex. In a series of papers, Adelson & Bergen (1983, 1985, 1986) have discussed a spatio-temporal energy model for human motion detection mechanisms which consist of linear filters that are oriented in space-time and tuned in spatial frequency. They have also explored and clarified the concept of spatio-temporal representation of motion, and highlighted the analogy between motion and orientation. Watson & Ahumada (1985) have applied this idea to propose a model of human visual motion sensing.

In contrast to the above biological models, there are a number of spatio-temporal methods for determining motion based upon the Marr-Hildreth theory of edge detection (Marr & Hildreth 1980). An early motion detector of this kind was proposed by Marr & Ullman (1981) who convolved images with the time derivative of the Laplacian of Gaussian. They applied this method to several real images and demonstrated its directional selectivity, but they could not determine the speed of motion. Buxton & Buxton (1983) have further extended the Marr-Hildreth theory by proposing that depth information could be extracted from the edges tracked over time using a spatio-temporal filter.

The spatio-temporal approach for detecting motion is now more attractive to a number of researchers in the computer vision community. Bolles and his colleagues have intro-

duced, and then generalised, the epipolar-plane image analysis for extracting structure from motion without determining motion (Bolles et al. 1987; Baker & Bolles 1989). This approach is also applied to the problem of optical flow. For examples, Heeger (1987a, 1987b, 1988) has tried to estimate optical flow using a set of spatio-temporal quadrature filters. Dickmanns & Graefe (1988a) have designed a dynamic monocular vision system using an integral spatio-temporal model that bypasses the non-unique inversion of the perspective projection by applying recursive least squares filtering. Liou & Jain (1989) have followed this approach and proposed a type of spatio-temporal inseparable surface model for motion detection, in which information about motion is obtained from parameters of the intensity hypersurface.

5.3 Motion as Orientation in the Spatio-Temporal Domain

A moving object may be viewed as occupying a 3D volume where there are two spatial dimensions, x and y, and one temporal dimension t. To analyse motion in the spatio-temporal domain, let us first consider a vertical bar moving to the right as shown in Fig. 5.1a. Its motion continuously sweeps out a 2D surface in space and time as shown in Fig. 5.1b. In the case of one-dimensional (1D) motion detection, it is assumed that motion occurs in one direction, say the direction x, and hence a moving object's trajectory can be described as a curve in the 2D spatio-temporal domain. For example, as the edge moves to the right, its motion trajectory can be pictured as a slanted strip as shown in Fig. 5.1c, which is a top view of Fig. 5.1b. In general, a 2D surface of this spatio-temporal solid volume indicates a motion

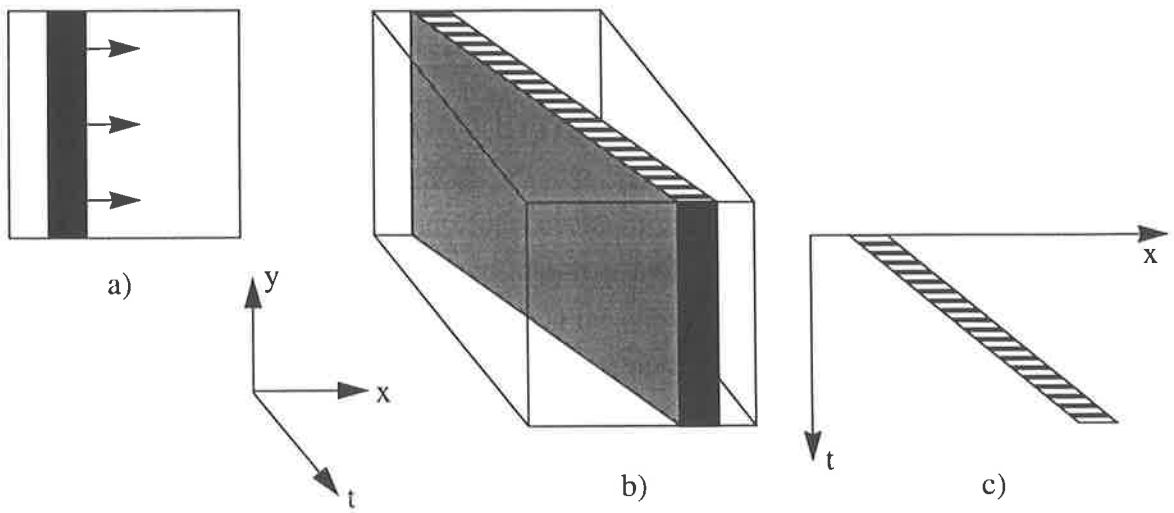


Fig. 5.1 a) A vertical bar moves to the right. b) Its corresponding 3D spatio-temporal image. c) A trajectory of a 1D motion as a top view of b) (Adapted from Liou & Jain 1986).

trajectory of a moving object, and the orientation of a tangent plane at every point on that surface indicates the speed and the direction of motion at that point (Liou & Jain 1989). Thus, the tangent plane (or tangent line in the case of 1D motion) reflects the velocity of the motion.

In summary, motion appears as orientation in the spatio-temporal domain. Therefore, the problem of detecting motion is equivalent to detecting the orientation of resulting tangent planes (or lines) in the space-time domain (Adelson & Bergen 1985; Liou & Jain 1989; Jähne 1993).

5.4 Motion Paths under the Template Model

In general, motion velocity varies with respect to time. However, motion can be considered to be uniform for a short period of time, that is, movement with a constant velocity for an appropriately small time interval. For the case of the 1D uniform motion of a bar, the motion trajectory of an edge can be represented as:

$$x = vt + x_0 \tag{5.1}$$

where x is the displacement, v is the velocity of the moving bar (or edge), and x_0 presents the initial position of the bar. Pictorially, this equation is represented as either the leading or the trailing edge of the slanted strip shown in Fig. 5.1c. If the motion depicted in Fig. 5.1c is temporally sampled, the result is a discrete stream of stimuli as shown in Fig. 5.2a. If the spatial axis is also quantized into spatial channels, and at each sampling instant the response of each channel is observed to determine whether or not an edge is present in that channel, then the exact position of an edge within a channel is not significant but what is important is to which channel an edge belongs at every sampling instant. In other words, the positions of a moving edge are detected at spatially quantized channels which are referred to as *detecting channels*. The detected positions of the moving edge at sampling instants are illustrated as dots in Fig. 5.2.b. By tracking the positions of these dots a motion trajectory can be formed in the shape of a staircase.

In fact, the technique of monitoring the position of a moving object at sampling instants is the nature of the template model, which has the visual field partitioned into channels (see Chapter 4). Each channel detects whether or not a moving edge belongs to it at each sampling instant by investigating changes in light intensity sensed by the channel. However, motion trajectories are not characterised by intensity change states, but instead by a number of templates since the templates rather than the intensity changes are observed.

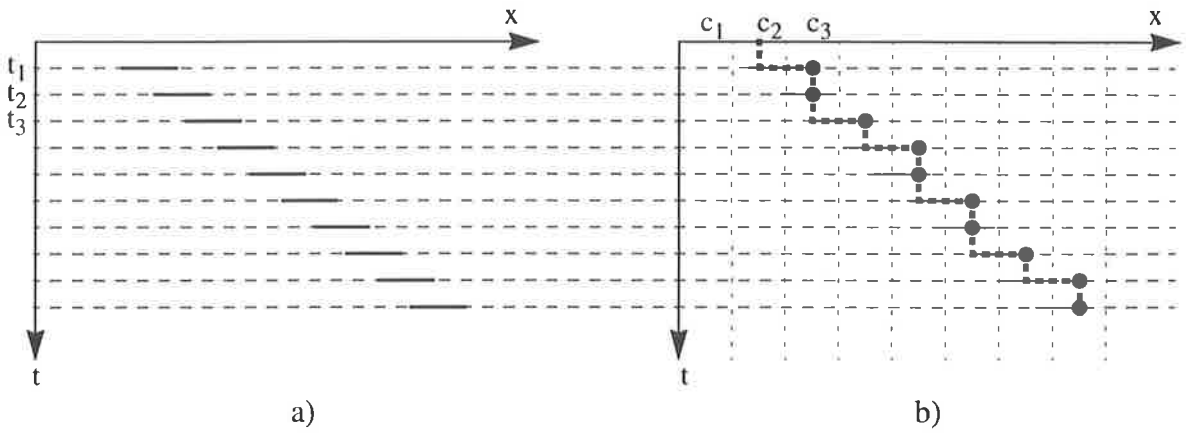


Fig. 5.2 a) Representation of a continuous motion of a bar, sampled in time. b) The positions of the leading edge of the bar are represented as dots within the spatially quantized channels called detecting channels. A staircase motion trajectory, the boldly dashed curve, is formed by connecting dots in the same channels until the leading edge is detected within another channel.

The technique of detecting the presence of moving objects in the spatial channels employed in the template model can be compared with the technique of sampling and holding signals, where the signals are sampled and their values are held constant until the next sampling instant. In the case of the template model, the positions of contrast edges are held in spatially discrete channels. The channel width is equivalent to the spatial quantization increment. It is obvious that the smaller the channel width is, the greater is spatial resolution, and the finer is motion detection. Similarly, if the sampling frequency is increased, the ability to detect fast motion is also increased.

Fig. 5.3 illustrates continuous and staircase trajectories of the uniform motion of an edge. On this staircase motion path, horizontal segments represent changes in space while vertical segments indicate changes in time. In other words, horizontal projections imply the distance an object moves in one sampling time interval, and vertical projections indicate the time an edge remains within the detection range of a channel. It is assumed that the motion of an object is small relative to the object's size, or the sampling frequency is sufficiently large, so that a motion trajectory is more likely generated by one object.

There are two kinds of corners: *concave* corners illustrated by shaded spots, and *convex* corners by black spots in Fig. 5.3. The corner concept is used in this study to pictorially clarify the staircase motion trajectory. These corner positions on a trajectory have special meaning to the movement of an edge, namely, the convex corners indicate a spatial transition between detecting channels, while the concave corners represent the current position of

the edge at the sampling instant. Geometrically, the concave corners are always above the next occurrence of a convex corner on the temporal axis. In other words, present concave and future convex corners occur in the same channel.

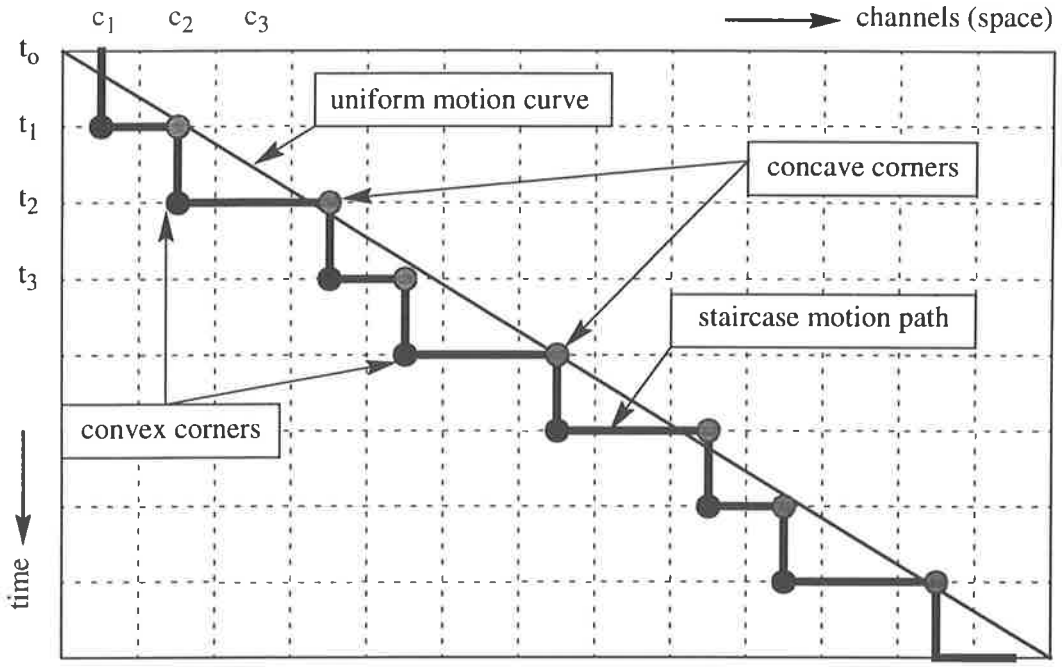


Fig. 5.3 Quantization of a motion trajectory using the template model. The solid line is a continuous motion trajectory of a uniform motion. The bold curve is a sampled motion path using the template model. Concave corners are illustrated by shaded spots and convex corners by black ones.

5.5 Feature Representation of Motion Trajectories

5.5.1 Templates on Motion Trajectories

When the template model was developed (see Chapter 4), it was claimed that directionally motion-sensitive templates occur at both convex and concave corners of a motion trajectory (see Fig. 4.2; Horridge 1990; Sobey & Horridge 1991). This claim is partially correct because the only templates occurring at convex corners are true DMSTs indicating intensity change in both the temporal and spatial axes. In contrast, templates occurring at concave corners have a single entry indicating change in intensity as shown in Fig. 5.4. The corresponding matrices of these templates have a zero determinant as shown in Table 4.1, and hence they do not qualify as DMSTs.

Furthermore, one intensity-change entry in these templates can be produced not only by motion but also by any random intensity fluctuations (i.e., noise) over the background. In other words, the templates occurring at concave corners are very sensitive to noise as discussed in Appendix C. On the other hand, the DMSTs may occasionally be induced by noise as will be shown in simulation results, but this happens rarely. In fact, experimental results show that DMSTs are robust templates indicating motion (see Section 4.7).

However, if the templates having one intensity-change entry and three no-change entries are purely generated by motion, they do occur at concave corners. In other words, they occur simultaneously in pairs with DMSTs at each sampling instant. They indicate the current positions of moving objects, and hence are referred to as *position conjugate templates* (PCTs). The PCTs indicate the locations of their DMST counterparts which occur at the next sampling instant. The pairs of DMSTs and corresponding PCTs are called *conjugate pairs* of templates, which are denoted by (DMST, PCT) occasionally. Thus, there are eight conjugate pairs corresponding to the eight DMSTs (Section 4.6.2). Similar to the display scheme of the DMSTs (Table 4.2), eight corresponding PCTs are encoded using numeric characters from '1' to '8'. Table 5.1 shows the eight conjugate pairs and their display codes.

Beyond those DMSTs and PCTs, there are other templates occurring on vertical or horizontal segments of a motion trajectory. Those templates lying on vertical segments, i.e., on the temporal axis, correspond to spatial contrast changes, and hence they are named *spatial templates.* Those templates occurring on horizontal segments, i.e., on the spatial axis, correspond to temporal contrast changes, and hence they are referred to as *temporal templates.* Fig. 5.4 illustrates the relationship between a motion path and the template occurrences on that path. The DMSTs are illustrated by shaded ovals at the convex corners of a motion path, while the PCTs are pictured by white ovals at the concave corners. Fig. 5.4a depicts a motion trajectory of *medium motion,* which is defined as the response obtained when an edge is moving at a speed of one channel per sampling interval. In this case the motion path consists of DMSTs and PCTs only. A faster movement will result in a path such as that of Fig. 5.4.b, which has longer horizontal segments where some temporal templates occur. Conversely, a slow motion is illustrated in Fig. 5.4.c, where some spatial templates occur on longer vertical segments.

In summary, there are four different kinds of templates occurring on motion trajectories:

- Directionally motion-sensitive templates, occurring at convex corners, indicate the spatio-temporal transition between the two adjacent spatial channels.
- Position conjugate templates, occurring at concave corners, indicate the current position of an object, and the next-occurring location of its DMST.

Motion to the right				Motion to the left			
Motion		Position conjugate		Motion		Position conjugate	
A		1		E		5	
B		2		F		6	
C		3		G		7	
D		4		H		8	

Table 5.1 Alphanumeric display codes of conjugate pairs.

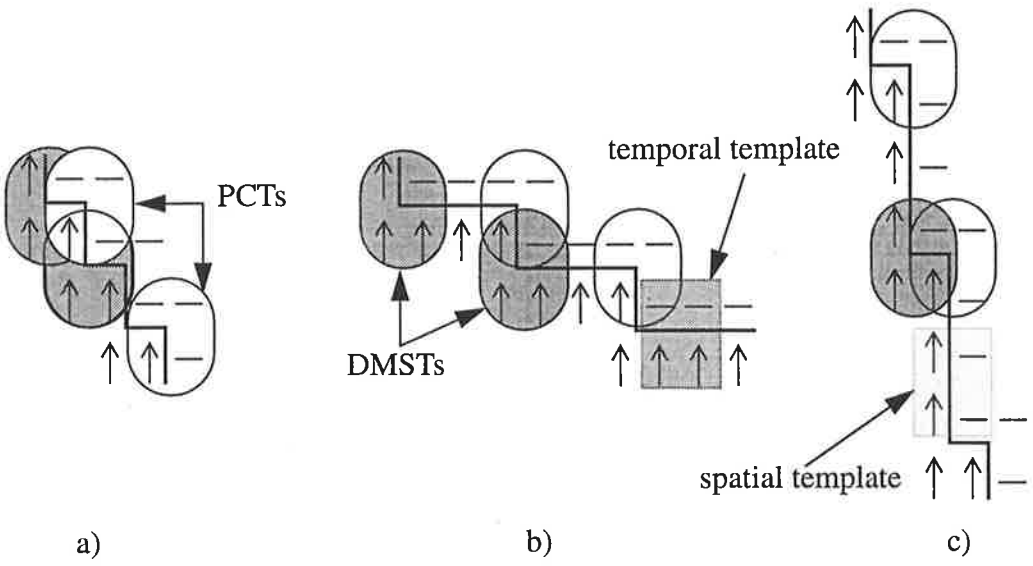


Fig. 5.4 Templates occurring on motion paths. Shaded ovals are DMSTs occurring at convex corners. White ovals are PCTs occurring at concave corners. a) Medium motion. Only DMSTs and PCTs on a motion path for a speed of 1 channel per sampling time period. b) Fast motion. There are some temporal templates (rectangular) on horizontal segments. c) Slow motion. There are some spatial templates (rectangular) on vertical segments.

- Temporal templates occurring on horizontal segments indicate fast motion.
- Spatial templates occurring on vertical segments indicate slow motion.

5.5.2 Scheme 1: Staircase Motion Trajectories using DMSTs and PCTs

Scheme 1, first mentioned by Nguyen et al. (1993b), is based upon the observation that a motion trajectory under the template model is a staircase path, on which a DMST and a PCT occur at convex and concave corners of the path respectively. As the DMST uniquely indicates a moving object's polarity and the direction of its motion, the pair of conjugate templates can characterise the motion trajectory of that moving object. Thus, motion trajectories can be represented as staircase paths using conjugate pairs of templates in the spatio-temporal domain. Thus, this scheme is referred to as *staircase representation.*

In order to completely represent a staircase motion path, this scheme requires that *a conjugate pair of templates must occur simultaneously at both corners of a motion path.* That is, the DMSTs must occur at convex corners and the corresponding PCTs must occur at concave corners of motion paths. However, this requirement may not be always satisfied, that is, a staircase motion trajectory may not be always continuous in the sense that some of its corners may not be recognised. This is referred to as *a problem of non-existence of complete staircase motion trajectories.* For example, in a noisy environment, some of the PCTs may not occur at concave corners of the trajectory as shown in Fig. 5.5 due to the noise-sensitive nature of PCTs (Appendix C).

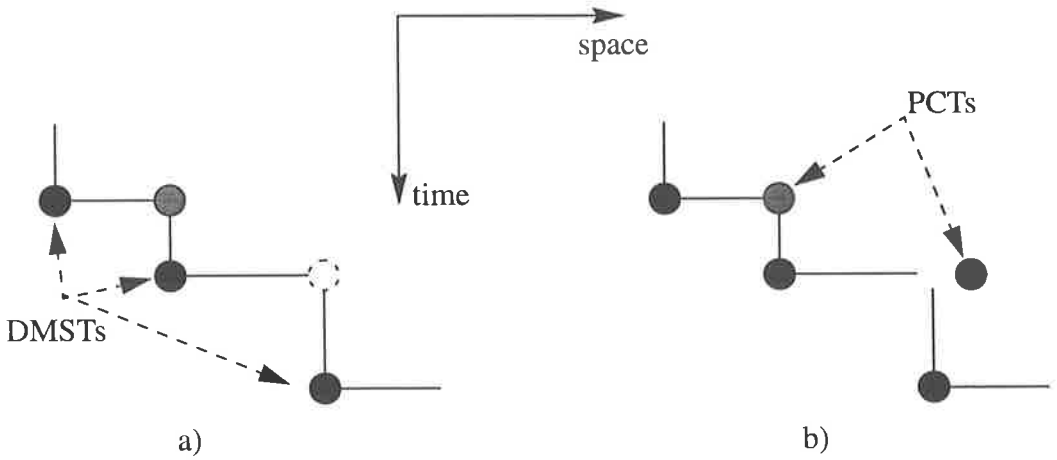


Fig. 5.5 The problem of non-existence of complete staircase motion trajectories since motion paths are not continuous due to: a) missing a conjugate template, and b) a conjugate template occurs at a wrong location due to noise. Black spots are DMSTs; shaded spots are PCTs.

In order to meet the above requirement, any noisy intensity fluctuations which are still contained in signals after the filtering should be removed so that there exist clear separations between no-change regions and intensity-change regions as illustrated in Fig. 4.11 and Fig. 4.12. Hence, a staircase motion path will be formed between a no-change and an intensity-change region. One way to achieve such clear separations is to increase thresholding levels to eliminate noisy fluctuations. However, as a trade-off, the increase in the threshold level will decrease the ability of the sensor to detect motion of low-contrast objects.

5.5.3 Scheme 2: Motion Trajectories using DMSTs as Features

If motion trajectories are determined completely, estimated velocities may be computed with a high accuracy. However, in some real-time applications involving fast reactions towards the environment highly accurate velocities may not be necessary, but quick estimates of velocity may be valuable. In such cases, motion trajectories are not necessarily identified before velocity information is quickly estimated.

One way to obtain quick velocity information is to quickly identify a number of locations of the object during the time course. That is, the object's motion trajectory is adequately represented with a number of points for the purpose of achieving quick estimates of velocity. This is the crux of scheme 2 of feature representation of motion trajectories. This scheme, first reported in Nguyen et al. (1995), is based upon the argument that velocity information can be obtained approximately if a number of points on the object's motion trajectory are known. As DMSTs occurring at convex corners of motion trajectories are robust templates indicating motion, they can be used as features for representing motion trajectories in the spatio-temporal domain as shown in Fig. 5.6. Hence, scheme 2 is referred to as *feature representation*.

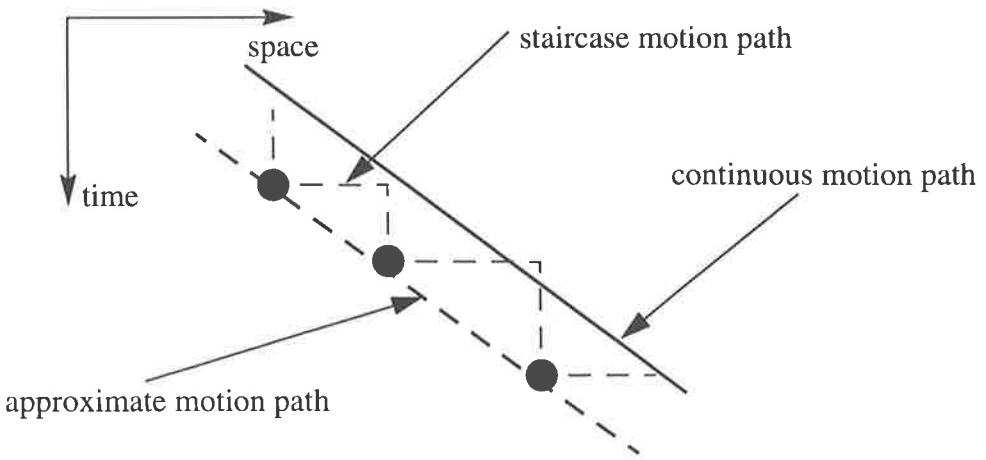


Fig. 5.6 A motion trajectory is represented using only DMSTs, illustrated by black spots.

5.5.4 Advantages

The two described schemes for feature representation of motion trajectories offer two benefits as discussed by Nguyen et al. (1995).

The first benefit is the data compression. This representation reduces the large amount of information associated with images to a minimum amount of necessary information for obtaining velocity information of moving objects. Conventionally, images, captured using a CCD camera for instance, may be represented using the gray-scale intensity with 8 bits per pixel. If all 81 templates are used for representing motion trajectories, 7 bits are required, hence not making any significant reduction. However, if only the DMSTs and their PCT companions are used as features for representing motion trajectories, only 4 bits per pixel may be needed. Thus, the reduction rate is 2 (i.e., $8 \div 4$). Although, this reduction is not great, the two discussed schemes prove, at least in principle, that in visual motion detection systems the data reduction can be achieved by representing motion trajectories using a number of features.

Hence, for four bits, there are 16 (i.e., 2^4) codes available for representing useful motion information. Obviously, one code is needed for representing all not used templates. The other 15 codes can be used for encoding feature templates.

With scheme 1, only up to 7 conjugate pairs can be used for representing motion trajectories due to the availability of 15 codes; each trajectory requires two codes. Thus, one of the eight conjugates pairs is not utilised. However, this does not cause any major drawback as the motion trajectory of one physical moving edge can be represented by two DMSTs (see Section 4.6.2), that is, by two conjugate pairs as a consequence.

With scheme 2, all the eight DMSTs can be used for representing all possible trajectories formed by moving objects.

The second benefit is that these representation schemes open up ideas for proposing novel tracking algorithms for velocity estimation (see Chapter 6 and Nguyen et al. 1993b). These algorithms require uncomplicated VLSI architectures when they are mapped into hardware (Nguyen el al. 1994b). This advantage is actually exploited when the sensor system is implemented in VLSI hardware (Chapter 8) as templates are encoded in such a way that each bit of their digital codes carries a piece of useful information such as motion, direction, polarity and so on. This hardwired template encoding will lead to a speedy processor whose operations are mostly based upon logical operations rather than on arithmetic operations (see Chapter 8).

5.6 Simulation Results

5.6.1 One Moving Object

The main purpose of the simulation of one moving object is to show different representations of the motion trajectory of an object which moves non-uniformly in front of the detector. In this simulation, the object is a bar moving from left to right and back in front of the sensor. The motion trajectory of the bar is modelled as a sinusoidal path.

Fig. 5.7 and Fig. 5.8 illustrate the staircase representation of the bar's motion trajectory at two different thresholding levels of 10% and 15% of the maximum intensity change †. The four conjugate pairs, (A,1), (B,2), (C,3), and (D,4), are used to represent four virtual paths moving to the right; the other four pairs, (E,5), (F,6), (G,7), and (H,8), are be used to represent those moving to the left. If the templates of the conjugate pairs are appropriately connected, staircase motion paths are obtained as marked in the two figures for the case of the pairs (D,4) and (A,1). However, these staircase paths may not be continuous since there are some concave corners where the templates '4' and '1' are missing as circled in Fig. 5.7 and Fig. 5.8 respectively. This is because of a noisy background which makes the position conjugate templates occur unexpectedly and randomly almost everywhere, for instance, as shown in Fig. 5.7. Fig. 5.8 shows that the position conjugate templates occur less randomly and frequently as the thresholding level is increased to remove most of noisy fluctuations in intensity. In other words, the position conjugate templates are sensitive to noise.

Fig. 5.9 and Fig. 5.10 show the corresponding feature representation of the bar's motion trajectory. The motion trajectory of the object moving to the right is recognised by four DMSTs 'A', 'B', 'C', and 'D', and that to the left is represented by 'E', 'F', 'G', and 'H'. For example, if curves are interpolated through the locations of the template 'A', virtual motion paths are obtained as drawn in the two figures.

Note that the thresholding level affects the occurrences of the DMSTs slightly only compared to the occurrences of the PCTs (see Appendix C). Occasionally, DMSTs occur on wrong places, for instance, marked DMSTs in Fig. 5.9 and Fig. 5.10. However, if the thresholding level is increased, those unexpected DMSTs are eliminated and the DMSTs do not occur any where except on the motion paths. This means that the DMSTs are robust templates indicating motion as shown in Fig. 5.10.

---

†. The maximum intensity change is searched for over a spatio-temporal image of intensity changes, then the thresholding level is set to be 10% or 15% of that maximum value.

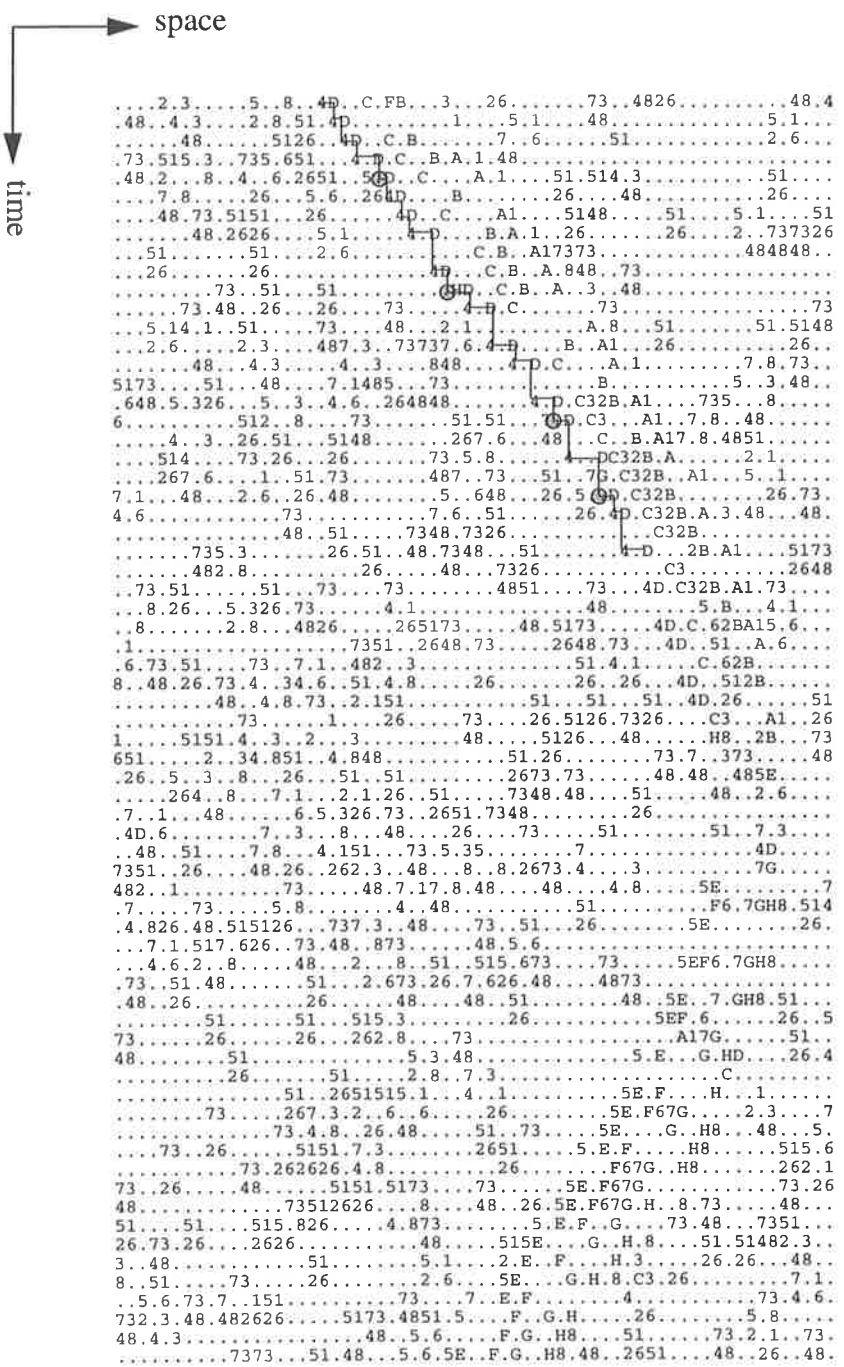


Fig. 5.7 Motion trajectory of a moving object using DMSTs and PCTs. SNR is 36.4 dB and thresholding level is 10% of the maximum intensity change.

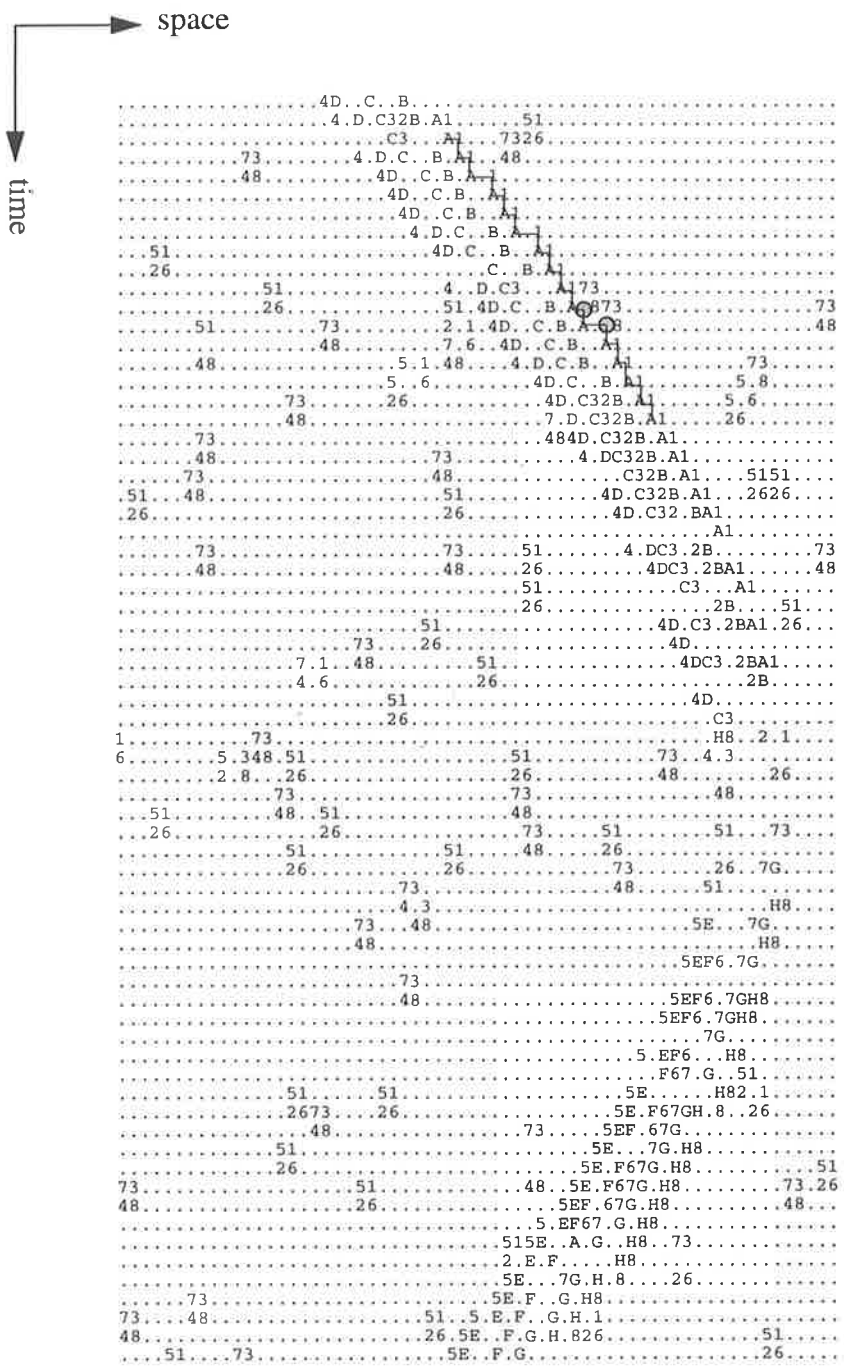


Fig. 5.8 Motion trajectory of a moving object using DMSTs and PCTs. SNR is 36.4dB and thresholding level is 15% of the maximum intensity change.

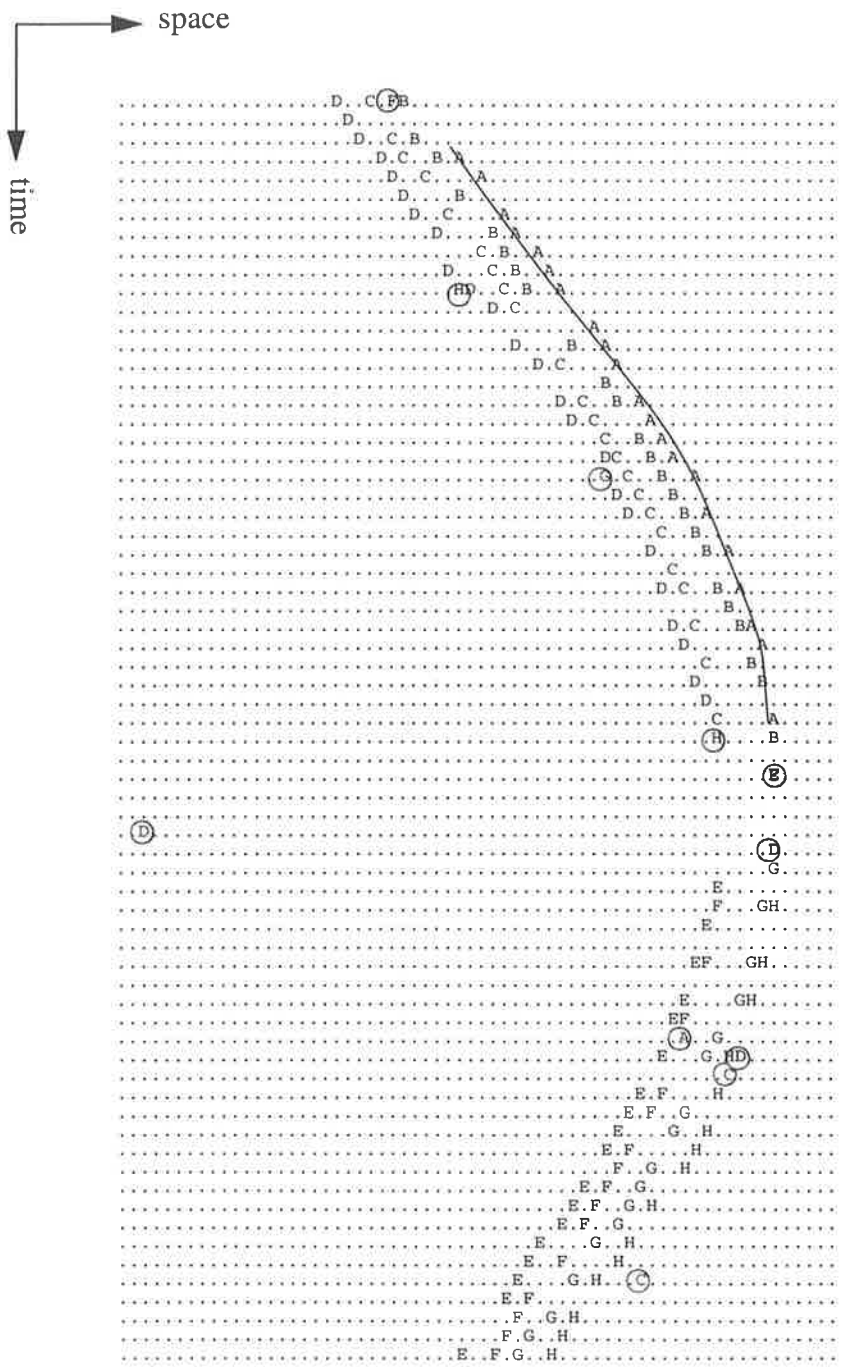


Fig. 5.9 Motion trajectory of a moving object using DMSTs only as feature points. SNR is 36.4dB and thresholding level is 10% of the maximum intensity change.

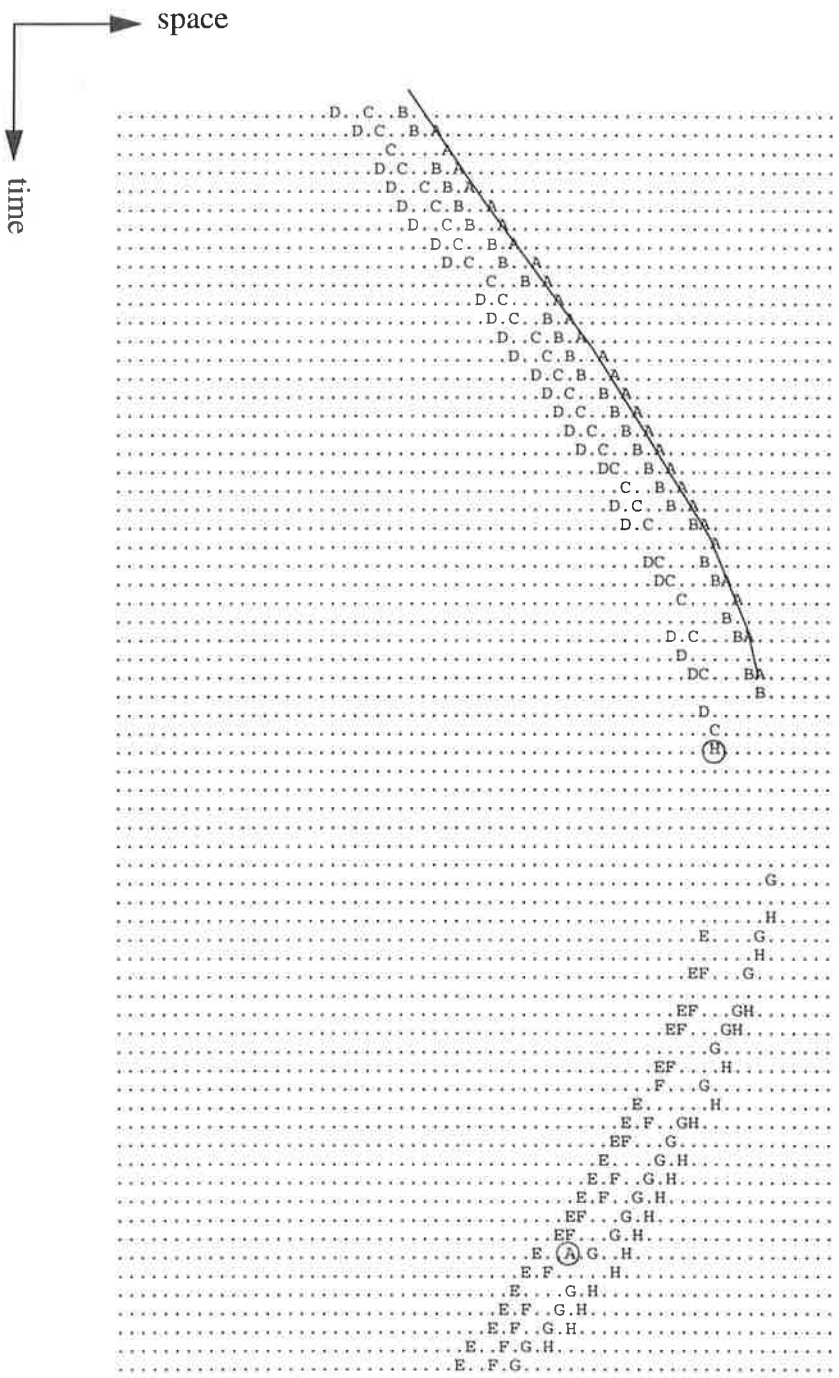


Fig. 5.10 Motion trajectory of a moving object using DMSTs only as feature points. SNR is 36.4dB and thresholding level is 15% of the maximum intensity change.

5.6.2 Two Moving Objects Occluding One Another

The primary objective of the simulation of two moving objects is to show how the template model can be applied to more realistic environments where there are often a number of objects which are simultaneously moving, and there may be objects occluded by other objects. The occlusion among objects usually creates difficult problems for detecting separate objects. Thus, it is worth to investigate the template responses at the intersection of motion trajectories in order to find out how the template model copes with such environments.

5.6.2.1 Crossing

A crossing situation occurs when two objects move in opposite directions and cross each other. In the vicinity of a crossing position (intersection of motion trajectories), the occluded object becomes the background scene for the occluding object. If the contrast between two objects is sufficiently high, the detecting channels can detect different levels of intensity changes between the two as shown in Fig. 5.11. As the bars have different luminance levels, their motion trajectories are distinguishable in this profile. Furthermore, the trajectory of the bar which is closer to the sensor is continuous at the intersection, while the further bar acts like the background for the closer at the intersection of their trajectories.

Fig. 5.12 and Fig. 5.13 illustrate the motion trajectories of the two bars using the two schems. In both the schemes, virtual motion paths are not continuous in the sense that the DMSTs rarely occur at the intersection. This discontinuity of the outer virtual paths can be explained as following. The nature of the template model is to form the templates from the intensity change states which are obtained by thresholding differentiated signals. In order to

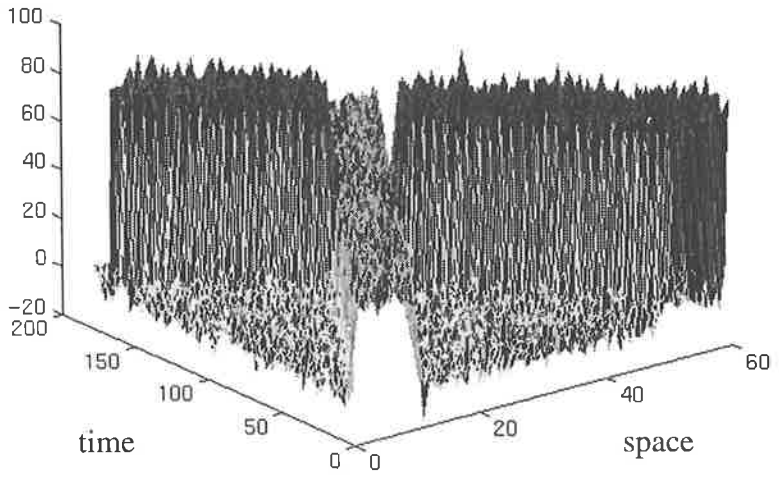


Fig. 5.11 A spatio-temporal intensity profile of two moving objects crossing one to another. Numbers on the vertical axis represent arbitrary units of intensity.

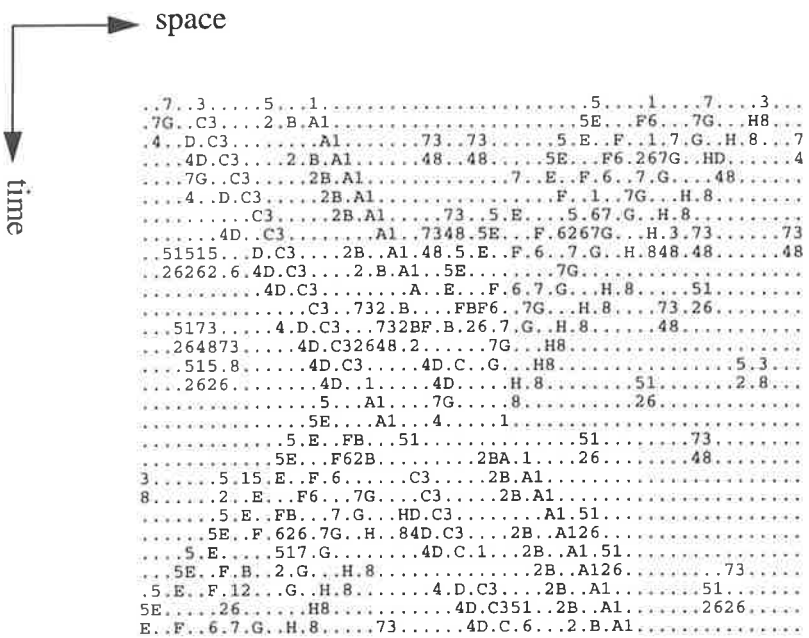


Fig. 5.12 Motion trajectories of moving objects, one crossing another, using DMSTs and PCTs.

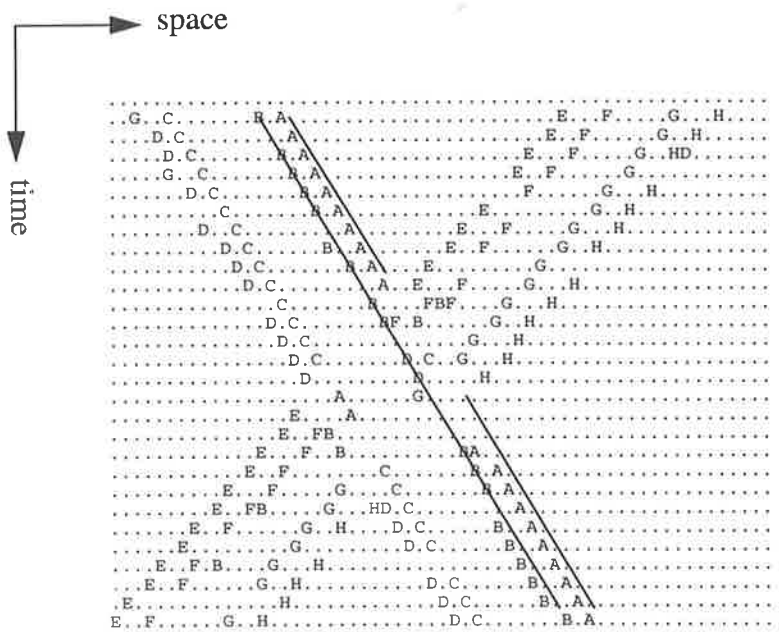


Fig. 5.13 Motion trajectories of moving objects, one crossing another, using DMSTs as feature points.

detect all objects sensed by the sensor, a selected threshold level must be lower than the lowest level of intensity changes caused by any of the objects present in the scene. As a result, at the intersection of the trajectories there is no sharp edge between intensity change (either decrease or increase) and no-change regions, as that generated by a moving object and a stable background respectively, as illustrated in Fig. 5.14. Consequently, DMSTs may not be expected at all at the intersection. Thus, at the occluding intersection, motion trajectories may not be completely represented by DMSTs and PCTs because an occluded object becomes the background, which may not be stable as a normally stationary background.

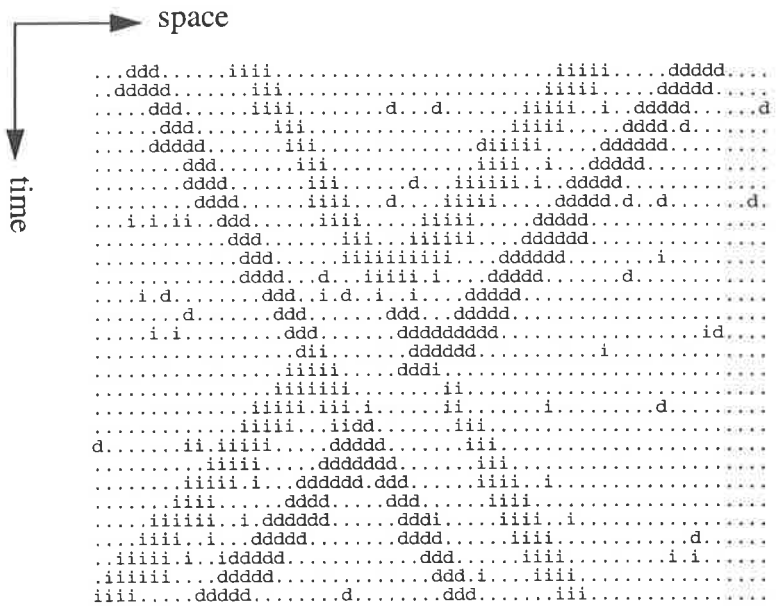


Fig. 5.14 Pattern of intensity changes at the intersection of a crossing situation.

5.6.2.2 Overtaking

Overtaking situation occurs when two objects move at different speeds in the same direction with the faster object overtaking the slower one. This situation becomes more interesting if the two objects have the same polarity since the template detector can not respond distinctly to movements of those objects. Hence, it is an issue when two objects which have the same polarity and are moving in the same direction are crossing one another.

Fig. 5.15 and Fig. 5.16 illustrates the motion trajectories of two such moving bars; one overtakes the other in both staircase and feature representation schemes. Similar to the crossing situation, in the vicinity of the occluding intersection, DMSTs occurring on the virtual paths are not present because the occluded object becomes the background for the occluding one.

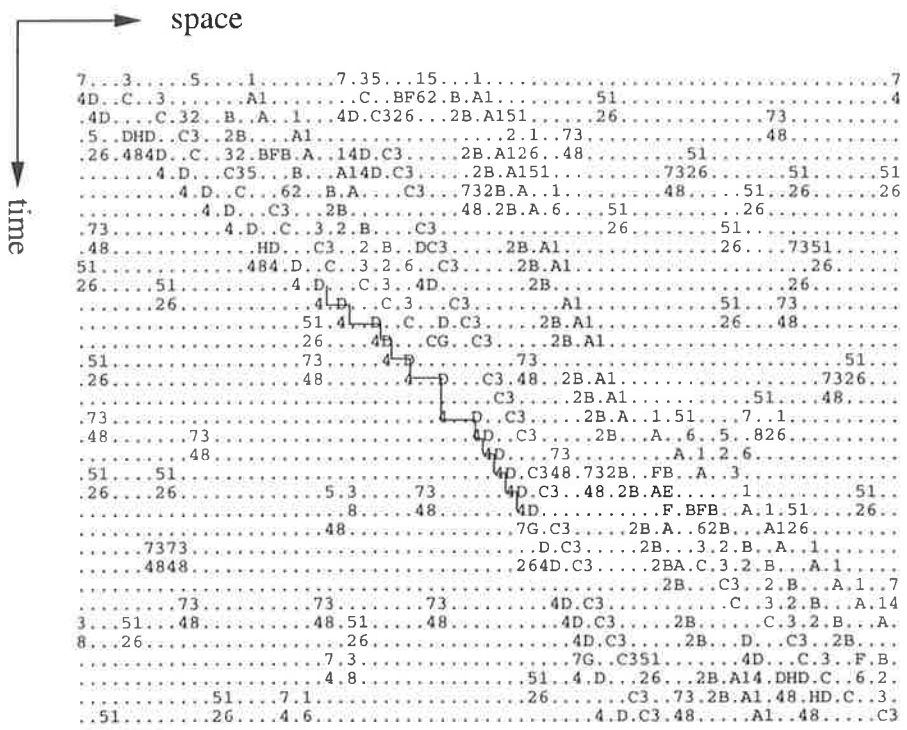


Fig. 5.15 Motion trajectories of moving objects, one overtaking another, using DMSTs and PCTs.

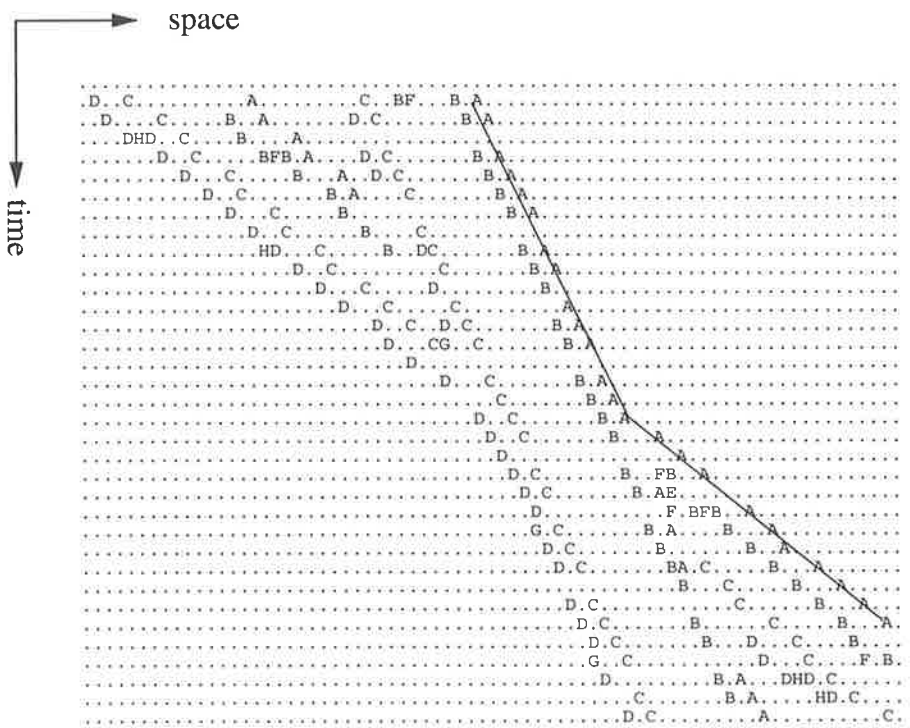


Fig. 5.16 Motion trajectories of two moving objects, one overtaking another, using DMSTs as feature points.

Furthermore, in either scheme, there is some degree of confusion between all the virtual paths of the two objects at the intersection due to the fact that all the paths are recognised by the same templates as the objects have the same polarity, that is bright-to-dark with respect to the background, and move in the same direction. For example the pair (D,4) forms a staircase outer virtual path as marked in Fig. 5.15, and the template 'A' forms a single non-linear outer virtual path as drawn in Fig. 5.16. This is a drawback of the template model.

5.7 Examples from Real-Time Experiments

In chapter 4, the motion patterns of a hand moving to the left and to the right were already presented in both the format of intensity changes and of the DMSTs (Fig. 4.11 & Fig. 4.12). Fig. 5.17 and Fig. 5.18 show motion patterns of two moving objects detected by

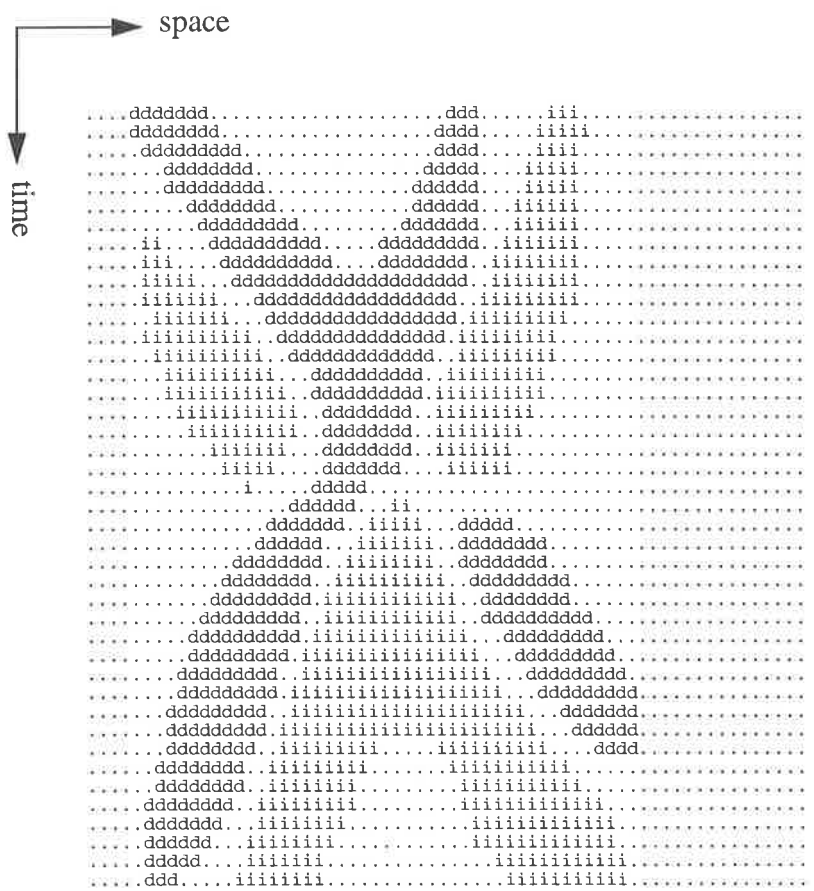


Fig. 5.17 A pattern of intensity changes at the intersection of motion trajectories of two moving objects detected by the smart micro-sensor II in real time. 'd' represents a decrease in intensity, 'i' represents an increase in intensity, and dot ('.') represents a no-change in intensity.

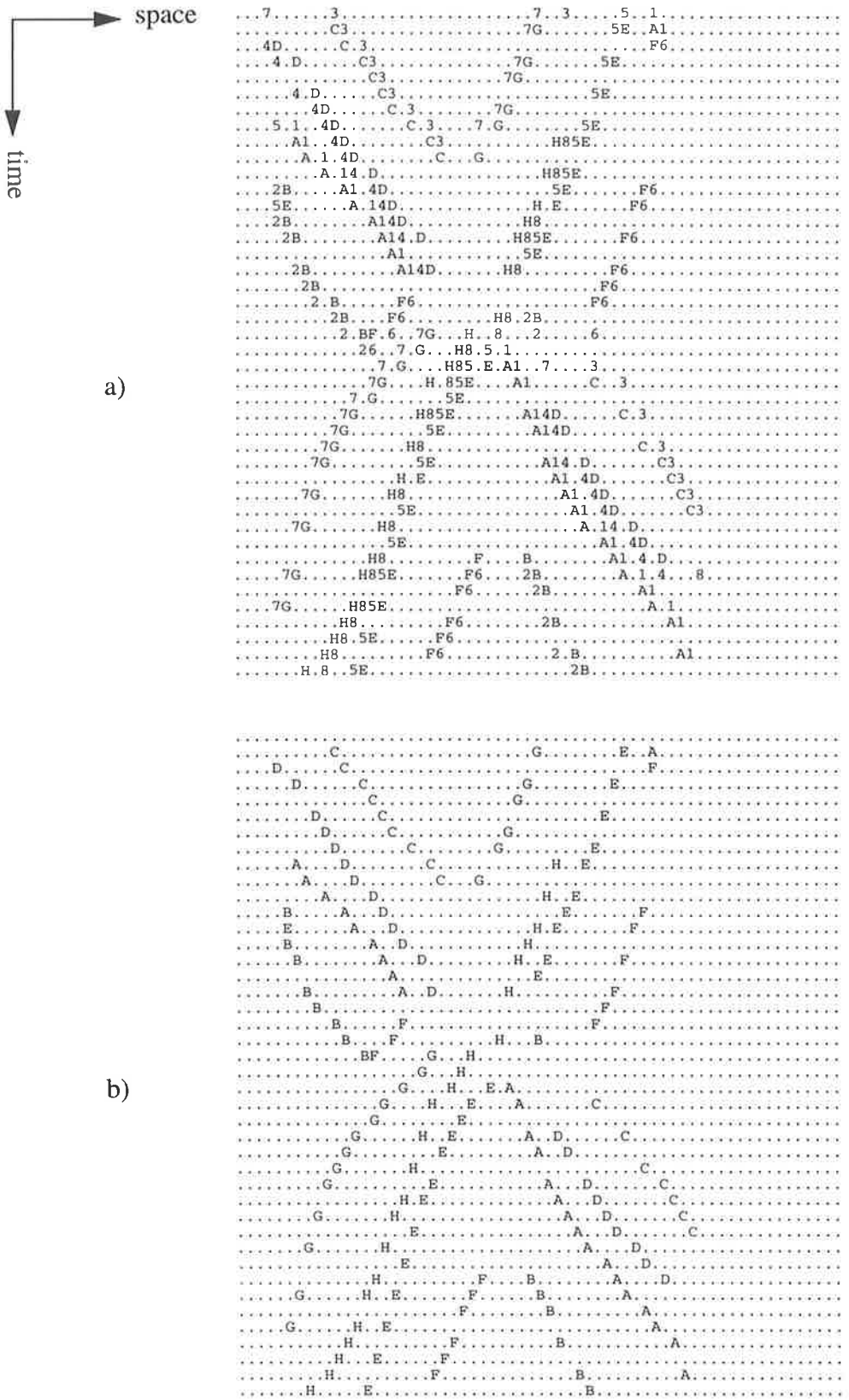


Fig. 5.18 Motion trajectories of two moving objects detected by the smart micro-senor II in real time. a) Staircase representation. b) Feature representation.

the smart micro-sensor II in real time. A closer object, at a distance of 30cm, is a bar moving from right to left and the further object, at a distance of 50 cm, is a hand moving in the opposite direction. Fig. 5.17 shows a pattern of intensity changes at the intersection of the motion trajectories of the bar and hand in real time. Fig. 5.18 shows their motion trajectories using both staircase and feature representation schemes. In this experiment, the background is virtually noise free, so it does not matter which scheme is used for representing motion trajectories. Real-time experimental results confirm that motion trajectories can not be clearly defined at the occluding intersection.

5.8 Summary

In this chapter we have provided a brief review of the spatio-temporal approach employed in motion detection and described how motion is considered as orientation in the spatio-temporal domain. We have also illustrated the staircase shape of motion trajectories on which there are four different types of templates occurring: directionally motion-sensitive, position conjugate, spatial, and temporal templates. However, a staircase motion path may be adequately represented using conjugate pairs of the DMSTs and the PCTs occurring at the convex and concave corners respectively. This scheme however is robust only in a virtually noise-free environment because the PCTs are sensitive to noise, and hence the PCTs are not well-behaved in a noisy environment.

In some applications where motion trajectories are not required to be completely determined, only the DMSTs can be used as features for representing motion trajectories in the spatio-temporal domain. This scheme is independent of the PCTs, hence is robust even in a noisy environment.

Experimental results of simulations and of real-time experiments show explicitly staircase motion paths and feature-pointed paths. The results also confirm that motion trajectories are not well defined at the occluding intersection of two or more objects.

The two discussed schemes, staircase and feature representation, offer two advantages. Firstly, the two schemes use only information related to coherent motion. In other words, an extraction of useful information is achieved. Hence, a data reduction is obtained. Secondly, the schemes open up ideas for proposing new tracking techniques for velocity estimation described in Chapter 6.

Chapter 6

Velocity Estimation

6.1 Introduction

Velocity information of moving objects is widely used in a number of applications. This information can contribute to scene interpretation such as recovering the structure of the environment (Huang & Netravali 1994), estimating time-to-contact or ranges for obstacle avoidance (Yakovleff et al. 1995; Ancona & Poggio 1993; Nelson & Aloimonos 1989), segmenting images (Potter 1977), and predicting future locations of moving objects (Chan et al 1994; Dickmanns & Graefe 1988b).

Despite its importance, velocity information can be either sufficiently estimated or accurately measured depending upon the application. This is referred to as the *Estimation versus Measurement* philosophy in the problem of visual motion detection (Ullman 1979a, 1979b, 1981; Prazdny 1980; Clocksin 1980). For example, for the navigational task an autonomous robot must detect all obstacles and then tries to avoid them but the robot does not need to know the exact shapes of all the objects surrounding it. The detection of objects can be carried out during the course of its movement and should be based upon low-level information rather than being based on the identification of objects or recovery of three-dimensional information (Anandan 1989). Therefore, estimation of visual motion is sufficient in the case of real time applications.

Following the feature representation of motion trajectories described in Chapter 5, velocity estimation is now a major concern, where different techniques may be used for the estimation. However, with the smart-sensing paradigm in mind, the simplicity in architecture and efficiency in processing time should be taken into consideration when suitable algorithms are proposed.

In this chapter we present the task of velocity estimation after moving objects have been detected by the template model. Section 6.2 shows the basics of velocity estimation. Section 6.3 presents three tracking algorithms, which have been proposed during the course of this research. Section 6.4 demonstrates some simulation results. A comparison among these techniques highlights some of their advantages and disadvantages in Section 6.5. The chapter closes with a conclusion.

6.2 Approaches for Velocity Estimation

In the spatio-temporal domain, if the motion trajectory of an object, described by Eq. (6.1), is known, its velocity can be determined using Eq. (6.2).

$$x = f(t) \tag{6.1}$$

$$v = \frac{dx}{dt} \tag{6.2}$$

However, it is not necessary to measure precisely the velocity as given by Eq. (6.2) in some applications. Instead, an estimated value which is given in Eq. (6.3) provided that the quantities Δx and Δt are small enough may be adequate. Eq. (6.3) shows that the velocity of a moving object (or edge) is estimated as the ratio of the displacement to the corresponding travelled time.

$$v = \frac{\Delta x}{\Delta t} \tag{6.3}$$

The analysis of the feature representation of motion trajectories in Chapter 5 suggests that there are at least two approaches for estimating velocity of moving objects from template data.

The first approach is based on an intuitive observation that convex (or concave) corners of a motion trajectory are approximately collinear in a short period of time. Hence, these convex (or concave) corners can be linearly interpolated, and the resultant slope is an approximate velocity of a moving object. Thus, the task of estimating velocities of moving objects can be reduced to the task of finding lines passing a number of DMSTs, which are used as features in spatio-temporal images. This task can be solved using the Hough transform (Sobey 1990; Sobey & Horridge 1990; Horridge 1991b). However, a common problem associated with the Hough transform is the computational cost (Illingworths & Kittler 1988).

The second approach consists of two phases. Firstly, motion trajectories of moving objects are established, and secondly, the moving objects' velocities are then estimated by Eq. (6.3). Two tracking algorithms for velocity estimation, which are originated from the two schemes of representing motion trajectories in the spatio-temporal domain (Chapter 5), have been developed. Using information on the direction of motion and the edge polarity, which is provided by the DMSTs, motion trajectories are tracked either by deterministically searching for their convex and concaves corners where DMSTs and PCTs occur respectively, or by locally and probabilistically searching for their convex corners where DMSTs oc-

cur. Hence, the task of estimating velocities of moving objects is equivalent to the task of tracking motion trajectories of those moving objects in a spatio-temporal image.

6.3 Tracking Algorithms for Velocity Estimation

6.3.1 Motivation

6.3.1.1 Efficiency in Processing Time

Efficiency in processing time is a vital issue in real-time systems which may require fast interactions towards the environment in order to accomplish tasks within a limited amount of time. In many real-time applications, a set of cooperating tasks are required to be solved. The tasks are often activated at regular intervals and must be completed by their deadlines (Shin & Ramanathan 1994). Thus, *time* is the most precious resource to manage at every stage in real-time systems. Our concern is in the time efficiency of the processing stage that can be achieved by implementing simple task-oriented algorithms.

6.3.1.2 Simplicity in Hardware Realization

Simplicity in hardware realization is the important constraint of limited computing resources in the smart-sensing paradigm (Chapter 2). The constraint of the VLSI hardware implementation level must be taken into account since the VLSI real-estate is limited. The more complex the algorithm to be mapped into silicon chips is the higher the total cost (including design time, silicon areas, and fabrication cost) of producing the chip will be. Thus, simple algorithms are preferable.

6.3.2 General Description

In order to exploit the benefits of using the staircase and feature representation schemes of motion trajectories provided by the template model without costly computational and hardware implementation phases, two tracking algorithms have been developed. These algorithms consist of tracking the motion trajectory of a moving object by searching for feature templates, which are either only DMSTs or with their PCT counterparts, in a small region of a spatio-temporal image of templates; velocity estimation is based upon a linear interpolation of the motion trajectory. However, these algorithms differ in a number of aspects which are discussed below.

The first tracking algorithm, called *stair-step tracking*, tracks motion trajectories by finding conjugate pairs of templates. Once a trajectory is established, the displacement and the corresponding travelling time can be measured along the trajectory by counting the numbers of the spatial and temporal contrast templates which appear on the trajectory of a mov-

ing object. The total number of temporal contrast templates on the horizontal segments of the path represents the distance the object travels while the total number of spatial contrast templates on the vertical segments measures the time taken. Thus, the velocity is computed as the ratio of the total number of temporal templates to that of spatial templates.

The second tracking algorithm, named *backward tracking*, searches for a particular DMST occurring at convex corners of a staircase motion trajectory backwards in time. Hence, the estimated velocity of a moving object is the velocity of that DMST measured as the ratio of its travelled distance to the corresponding taken time.

In parallel to the development of the two aforementioned tracking algorithms, there is another algorithm, named *forward tracking*, proposed by Yakovleff et al. (1994). This technique is similar to the backward tracking algorithm in the sense that it searches for a certain kind of feature template only, either DMST or PCT, but differs from the backward tracking mainly by predicting a future occurrence of a feature template in a spatial vicinity of the present response of that feature template. As a result this algorithm has a completely different hardware architecture from those of the stair-step tracking and backward tracking algorithms (Yakovleff et al. 1994; Yakovleff 1995).

It is worth noting that these algorithms are not modelled using the probability theory, or by mathematical equations, but are based upon empirical and analytical observations of template responses to motion. These techniques assume that feature templates are present in images if there is motion, and their task is only to search for the feature templates. They do not attempt to infer the presence of these templates if some of the feature templates are missing due to noisy conditions. In other words, the problem of searching for feature templates in a spatio-temporal image is a binary problem as a feature template is either found or not. If a feature template is found, its velocity can be evaluated regardless of other templates. If not, it is assumed that there is no motion.

In principle, all the three tracking algorithms detect the occurrences of feature templates (either DMST, PCT, or both of them). The estimated (average) velocity is then computed as the ratio of the displacement between the first and last occurrences of a DMST, to the number of sampling intervals in between. In other words, a linear approximation for speeds is carried out. The accuracy of the estimated velocity depends mainly on how long a motion trajectory is tracked, i.e., the memory of the tracking. In fact, a memory which accommodates 8 images obtained at 8 consecutive sampling instants is proposed as a compromise among the following conditions:

- As the template model involves the spatial quantization of the visual field into detecting channels, quantization errors affect the accuracy of the estimated velocity. The analysis in Appendix E shows that, given a resolution of the spatial quantization, the error of the estimated velocity of an object moving uniformly is inversely proportional

to the number of accumulated images over which the motion trajectory of the object is determined. So, a large number of images should be accumulated in order to increase the accuracy of the estimated velocity. Consequently, a large memory is required to accommodate more template data.

- In contrast, in the case of non-uniform motion, estimated velocities which are linearly approximated over a short period of time (i.e., a small number of accumulated images) may be more accurate that velocities approximated over a long period of time, for instance, as shown in Fig. 6.1. The linearly approximated velocity through *A* and *C* is closer to the true velocity (i.e., the slope of *AA'*) than the linearly approximated velocity through *A* and *B*.

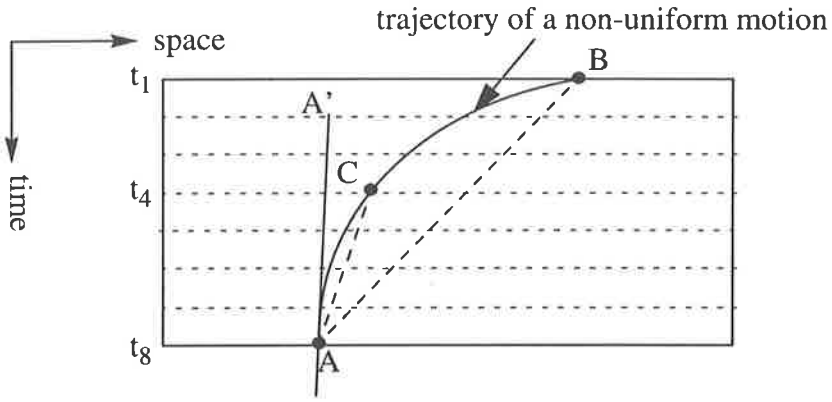


Fig. 6.1 Linear approximation of speeds for non-uniform motion.

- Furthermore, memory capacity is a function of hardware constraints and simple hardware is one of the objectives of the research (see 6.3.1.2), and hence a small size would be desirable.

6.3.3 Search Space

A search space is defined as a spatio-temporal region over which the tracking algorithms calculate a range of possible velocities. Based on experimental results (see Chapter 8), such possible velocities are set to be lying between the maximum velocity of 18 channels (either to the left or to the right of a reference channel) per 7 sampling intervals (i.e., between 8 sampling instants) and the minimum of 2 channels per 7 sampling intervals as shown in Fig. 6.2. This search space is applied to all three tracking algorithms.

Hence, for a sampling rate of 100 Hz and each channel covering a 1 degree of the visual field, the possible velocities range from 28.6 degrees per second to 257.1 degrees per second.

To calculate the minimum velocity, a feature template must be searched for over at

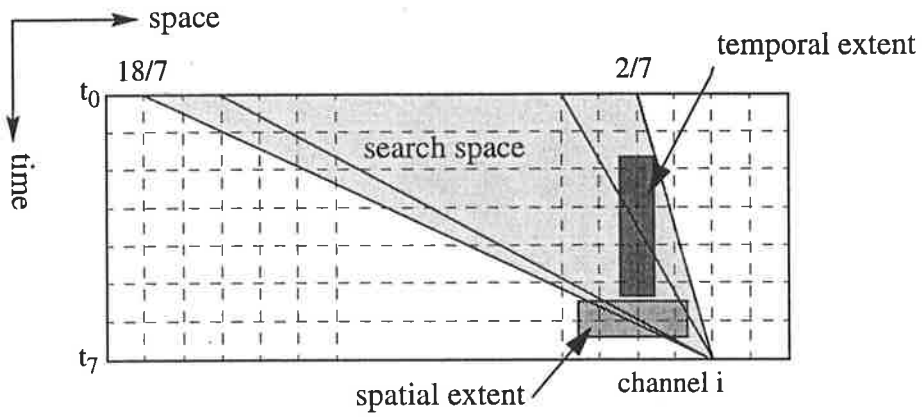


Fig. 6.2 Search space.

least 4 sampling instants within one channel. However, to ensure that a search is being carried out within the search space only, a constraint of 4 sampling instants is also imposed to the search along the time axis (or vertical search), that is, the vertical search for a feature template with one channel is carried out for 4 sampling instants only. If the feature template is not found within 4 sampling instants, the feature template is assumed to be not occurring. This constraint is called the *temporal extent T* as shown in Fig. 6.2.

To calculate the maximum velocity, a motion template must be searched for across at least 3 channels at each sampling instant. LIke the temporal extent, a constraint of 3 channels is also imposed to the search along the space axis (or horizontal search), that is, at each sampling instant the horizontal search for a feature template is carried out across 3 channels only. If the feature template is not found within 3 channels, the feature template is assumed to be not occurring. This constraint is called the *spatial extent S* as shown in Fig. 6.2.

6.3.4 Stair-Step Tracking (SST) Algorithm

The stair-step tracking algorithm, first reported by Nguyen et al. (1993b), was developed to exploit the benefits provided by the first scheme of representing motion trajectories as staircases. It is primarily based upon the premise that a staircase motion trajectory of a moving object under the template model can be identified by a conjugate pair of templates (i.e., a DMST and a corresponding PCT as shown in Table 5.1) which occur at the convex and corners of the motion trajectory, respectively. Hence, the conjugate pair can be tracked in order to identify the trajectory of that moving object. The conjugate pair acts like turning indicators at the corners of the trajectory, which guide a tracker to follow the trajectory correctly.

Initially, the tracker has to find a DMST as a starting point on a trajectory, then it moves vertically (i.e, along the temporal axis) to search for the corresponding PCT. When the corresponding PCT is found, the tracker then moves horizontally (i.e., along the spacial

axis), either to the left or to the right depending upon the direction of motion provided by the DMST, searching for the DMST again. By following this procedure, the motion trajectory of any moving object can be tracked. Thus, the algorithm preforms a deterministic search for motion trajectories.

This algorithm can be used either forward or backward in time. *Forward-in-time* tracking means that the algorithm tracks a motion trajectory from a point in the past to the present and vice versa for *backward-in-time*. In this study, the *stair-step backward-in-time tracking* algorithm is described as follows:

Initially, find a DMST as starting point.

Do loop over 7 sampling instants

- *If a DMST for motion to the right (left) is found at a convex corner, then track upwards until the corresponding PCT is found at a concave corner.*
- *If a PCT for motion to the right (left) is found at a concave corner, then track horizontally to the left (right) until the corresponding DMST is found at a convex corner.*

The principles of this algorithm are illustrated in Fig. 6.3. The staircase motion trajectory of a moving object shows the movement of the object to the right. The trajectory is not

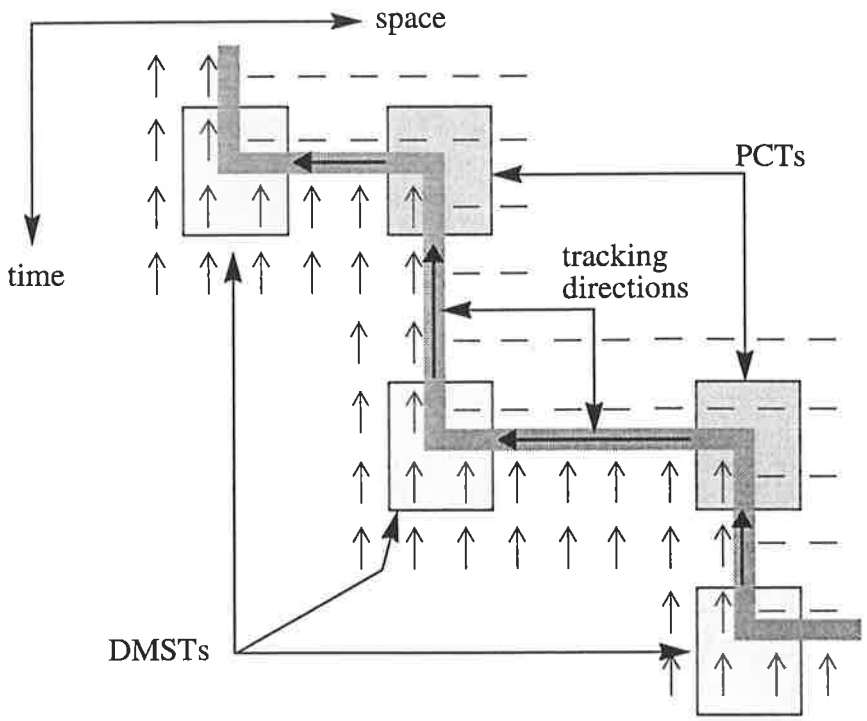


Fig. 6.3 Stair-step tracking strategy.

regular since motion, in general, is not uniform. The thick arrows indicate the tracking direction, and this is a backward-in-time technique as the tracking direction is opposite to the time and motion direction.

Although the stair-step algorithm uses a conjugate pair of templates (DMST, PCT) to track a motion trajectory in 8 sampling instants, the computation of the average velocity is based upon the first and last occurrences of the DMST. The algorithm calculates only velocities within the defined search space. This means that the temporal extent T is applied to the vertical search for PCTs and the spatial extent S is applied to the search for DMSTs on the horizontal direction.

6.3.5 Backward Tracking (BT) Algorithm

The backward tracking algorithm was developed to calculate velocities of moving objects whose motion trajectories are represented using the DMSTs only as feature points, i.e., using the second scheme of representing motion trajectories. This algorithm is based upon the argument that if a DMST currently occurs at a certain position, indicating the direction of motion, the same DMST must have previously occurred in its neighbourhood in the opposite direction. Without using the PCTs, a motion trajectory cannot be completely followed. However, if DMSTs are present, a search for these templates as targets can be carried out. A search window is set up in the neighbourhood area of a currently detected DMST in a previous image. Then a search for the target DMST is performed within the search window.

The principles of this algorithm are demonstrated in Fig. 6.4 and its procedure is as follows:

1. *In the image at the current time* t_n *search for a DMST at a position P.*

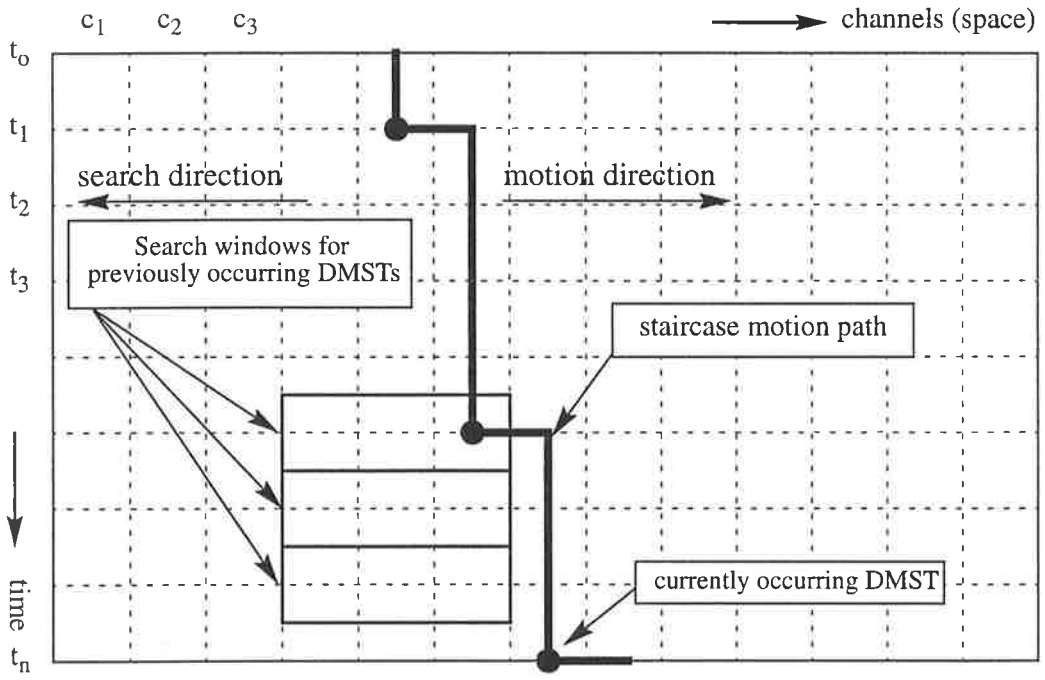
2. *In the image at the time* t_{n-1} *a search window is established a priori, to the left or the right of P depending on the direction of motion indicated by the DMST. The size of this search window is the spatial extent S (i.e., 3 channels) in order to ensure the search is being carried within the define search space.*

3. *Search for the same DMST in the search window.*

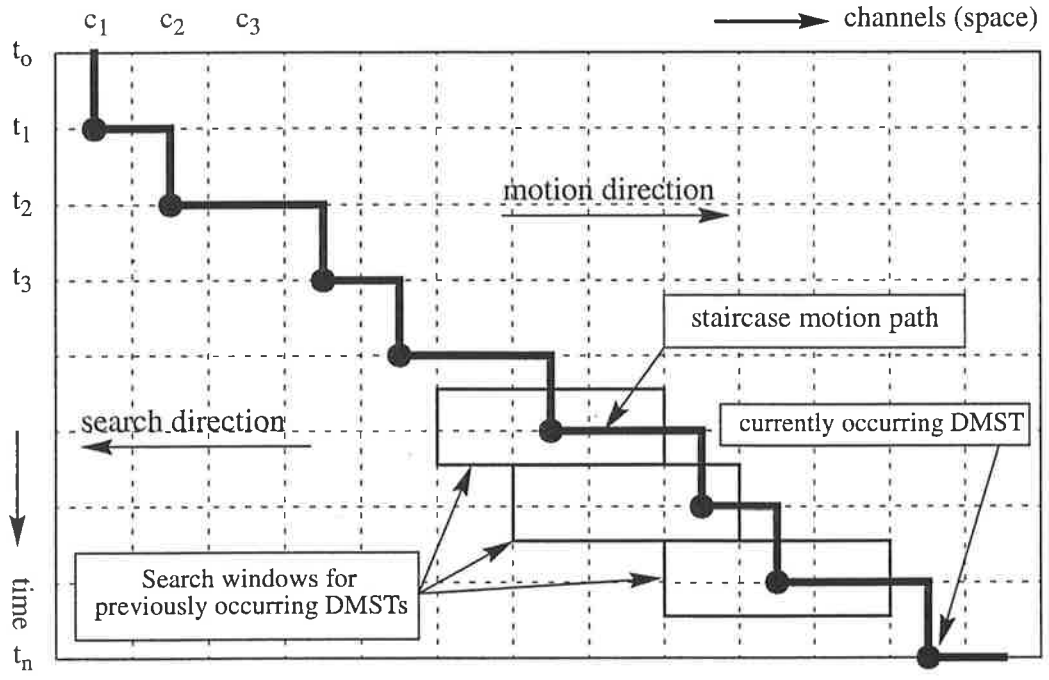
a. *If the target is found, the location of the currently found DMST becomes the position P, and go back to step 2.*

b. *Else go to step 4.*

4. *Step 2 is applied to the image at the time* t_{n-2} *as illustrated in Fig. 6.4a, that is, the DMST is searched for over a number of times (not larger than the temporal extent T of 4 sampling instants) to calculate a velocity for a slow motion.*



a)



b)

Fig. 6.4 Backward tracking strategy for moving objects. a) Slow motion. b) Fast motion.

In this technique, the task of searching for the DMSTs in a spatio-temporal image can be considered to be equivalent to the task of correlating signals in the correlation technique, which has been widely applied in the computer vision community for motion detection.

6.3.6 Forward Tracking (FT) Algorithm

Tracking for feature templates can be done in the forward direction as well, in which the hardware realization can be simplified and processing accomplished in parallel. The forward tracking algorithm is based upon the premise that if a feature template has responded at a certain position, indicating the direction of motion, then the same moving object will cause an identical template to eventually occur nearby, that is, a few detecting channels further in the direction of motion. In other words, if a feature template is currently detected at a location, the same template will probably be detected in a neighbourhood of that location in the direction of motion. In order to ensure that the search is being carried out within the defined search space (Section 6.3.3), at each sampling instant a search window, whose size is 3 channels (i.e., the spatial extent), is set up in the vicinity of the previous occurrence of the feature template, and the search for the feature is carried out within the window. The principles of this algorithm are illustrated in Fig. 6.4.

The motivation of this algorithm is to avoid the problem of storing a sequence of images in a memory and tracking for motion trajectories. Instead, this technique searches for a

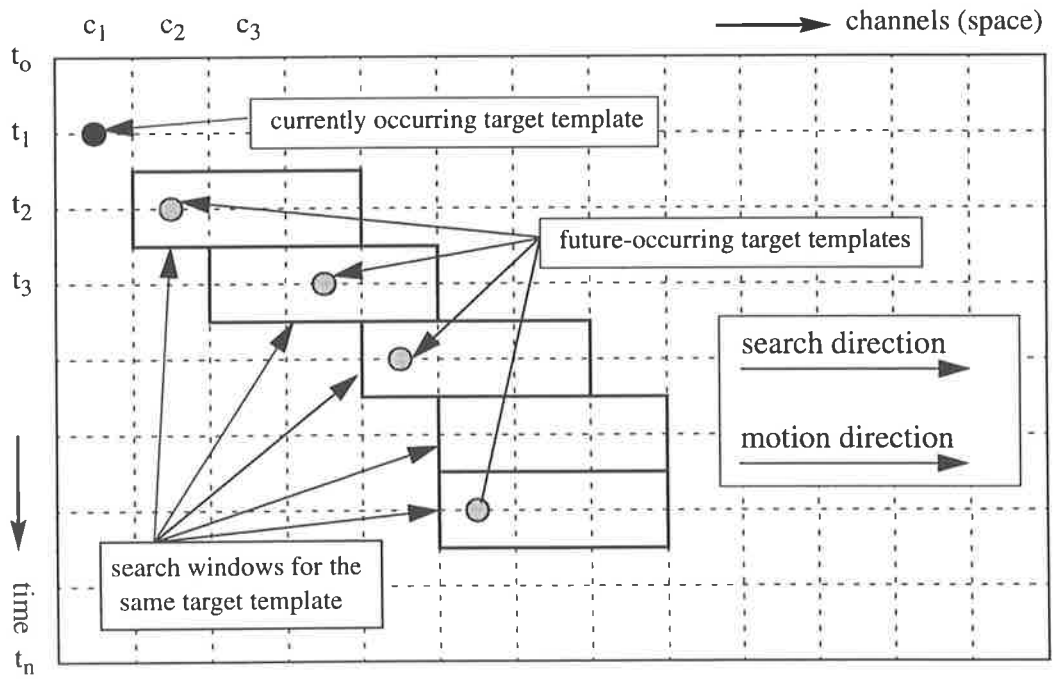


Fig. 6.5 Forward tracking strategy for moving objects.

known feature template in the vicinity of previously occurring identical one. It keeps track of the past history of previous positions of a feature template for 8 sampling instants (i.e., the temporal size of the memory).

The mapping in hardware of the forward tracking algorithm consists of a number of tracking "engines" operating independently from one another. The operation procedure of the tracking engine is described by Yakovleff et al. (1994) as follows:

Each time the target(ed) template is found within a specific catch range and direction of motion, its position is stored as the current position, and its displacement relative to the previous current position is shifted into a first-in-first-out (FIFO) register of length L † *(i.e, 8 sampling instants). As the FIFO is shifted at each iteration, the sum of its contents represents the total displacement D of the template during time L, yielding velocity D/L.*

In the case of slow motion, the object may take some time steps to move from one channel to the next; that is, the feature template may occur once in a few sampling instants. In this case, once a tracking engine is engaged with a feature template, and if it does not find the feature template again, it assumes that motion is slow, and the engine shifts a zero displacement into the FIFO register and reports an estimated velocity. However, this assumption makes the tracking engine continuously report velocity estimates even though objects may have stopped moving or have changed their heading direction. To avoid this problem, the temporal extent T (i.e., 4 sampling instants) is also constrained on the vertical search such that if the engine does not find the feature template for 4 sampling instants, it concludes that the object is not moving in that direction any more and the tracking engine stops functioning.

6.4 Simulation Results

This section is devoted to the experimental evaluation of the stair-step, backward and forward tracking algorithms described in the previous section. Experiments described in this section are mainly quantitative, and are divided into three cases as follow:

1. Uniform motion of one object.
2. Non-uniform motion of one object, and
3. Two objects moving simultaneously.

The general objective of these experiments is to quantitatively judge the correctness of angular velocities recovered by the algorithms, and to analyse their behaviour with respect to various movements of objects encountered. To fulfil this goal, the true motion of

†. T was originally used in Yakovleff et al. (1994). However, T is denoted for the temporal extent, hence L is used instead.

objects must be known. Typically, it is possible to know in advance the motion of an object only if the motion is synthetically generated using either software or hardware. In these experiments, template data are generated using the technique of data acquisition as described in Appendix B. Real-time experiments where movements of objects are detected using the smart micro-sensor II will be described in Chapter 8.

In the first two simulations, template data of one moving object with known velocities are created in order to quantitatively verify the performance of the SST, BT, and FT algorithms. The third simulation is dedicated to a more realistic dynamic scene in order to gauge how the tracking algorithms cope with a complex environment where some objects may be occluded by others.

6.4.1 Uniform Motion

6.4.1.1 Error Analysis

In the simulation of uniform motion, a bright object is moving cross the visual field of the detector with a known constant velocity. Its motion trajectory is represented in Eq. (6.4), where v is its velocity of 230 detecting channels (or degrees) per second, x_0 is an arbitrary initial position and t is time.

$$x = vt + x_o \tag{6.4}$$

From the analysis in Appendix E, the theoretical maximum error due to the spatial quantization is given by

$$\text{maximum quantization error} = \frac{100 \times \text{channel size}}{\text{velocity} \times \Delta T} \quad \%$$

where channel size is 1 degree, and ΔT is the time taken by the object to moving from channel to another. Hence, for a velocity of 230 degrees per second and ΔT of 35 milliseconds (i.e., 7 sampling intervals of 200 Hz), the maximum quantization error is 12.42%.

6.4.1.2 Results

Fig. 6.6 shows the corresponding image of templates. The virtual motion trajectories are recognised by the conjugate pairs (A,1), (B,2), (C,3), and (D,4), where the pair (A,1) indicates movement to the right of the object's leading edge. This pair is used as the target templates in the tracking algorithms.

The SST algorithm tracks the pair (A,1), while the others calculate the velocity of the object on the basis of movement of the DMST 'A'. Fig. 6.7, Fig. 6.8, and Fig. 6.9 show the

simulation results using the SST, BT, and FT algorithms respectively. The estimated velocities and relative errors using each algorithm are plotted in the same figure.

The estimated velocities obtained from both the BT and SST algorithms are identical in this case since the motion trajectory of the object is well established by the conjugate pair (A,1). The relative error is very small for most of the time, and in fact, the minimum error is 0.62%. Although there are some instants when the error rises to the maximum of 13.04%, overall the two algorithms can recover the velocities of the object from a spatio-temporal image of templates. In Fig. 6.7 and Fig. 6.8, there is a gap where no velocity information is available because no template 'A' was detected at the fourth row after its first occurrence.

The performance of the FT algorithm is almost the same as that of the SST and BT algorithms, except that the result of this algorithm fills the gap presented in Fig. 6.7 and Fig. 6.8 with a rough estimate (i.e., above 40%), and that the FT results in some less accurate

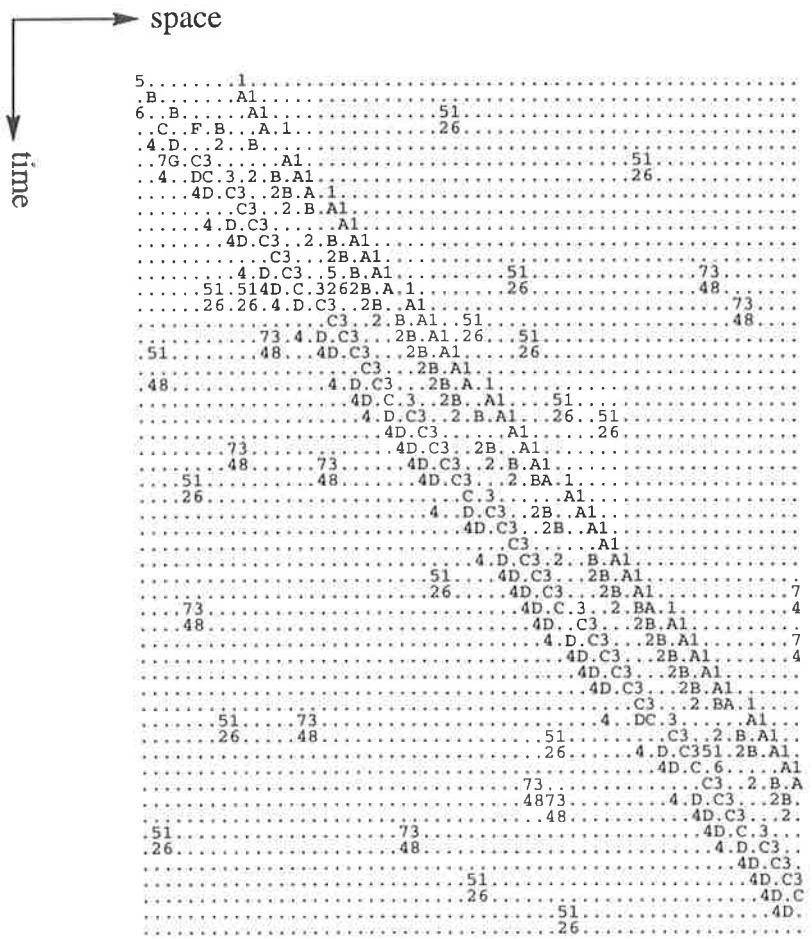


Fig. 6.6 A part of the spatio-temporal image of templates created by a uniform motion of an object, whose motion trajectory is described as a linear equation. The motion of its leading edge is recognised by the conjugate pair of templates (A,1).

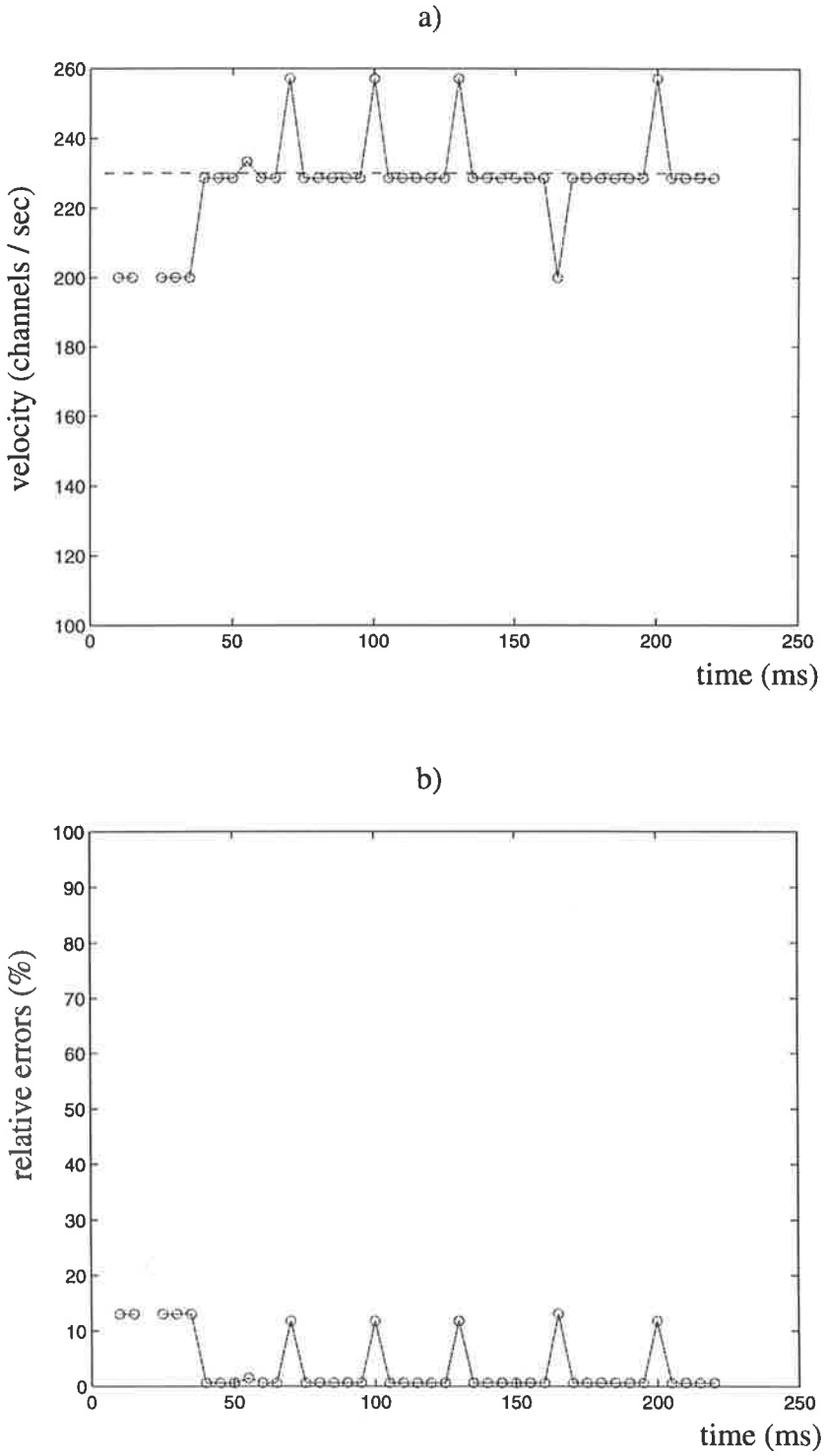


Fig. 6.7 The velocity estimation using the SST algorithm. a) The dashed line represents the true velocity of the moving object. The solid curve passing through symbols ('o') is the estimated velocity. b) The magnitude of relative errors of the estimated velocity.

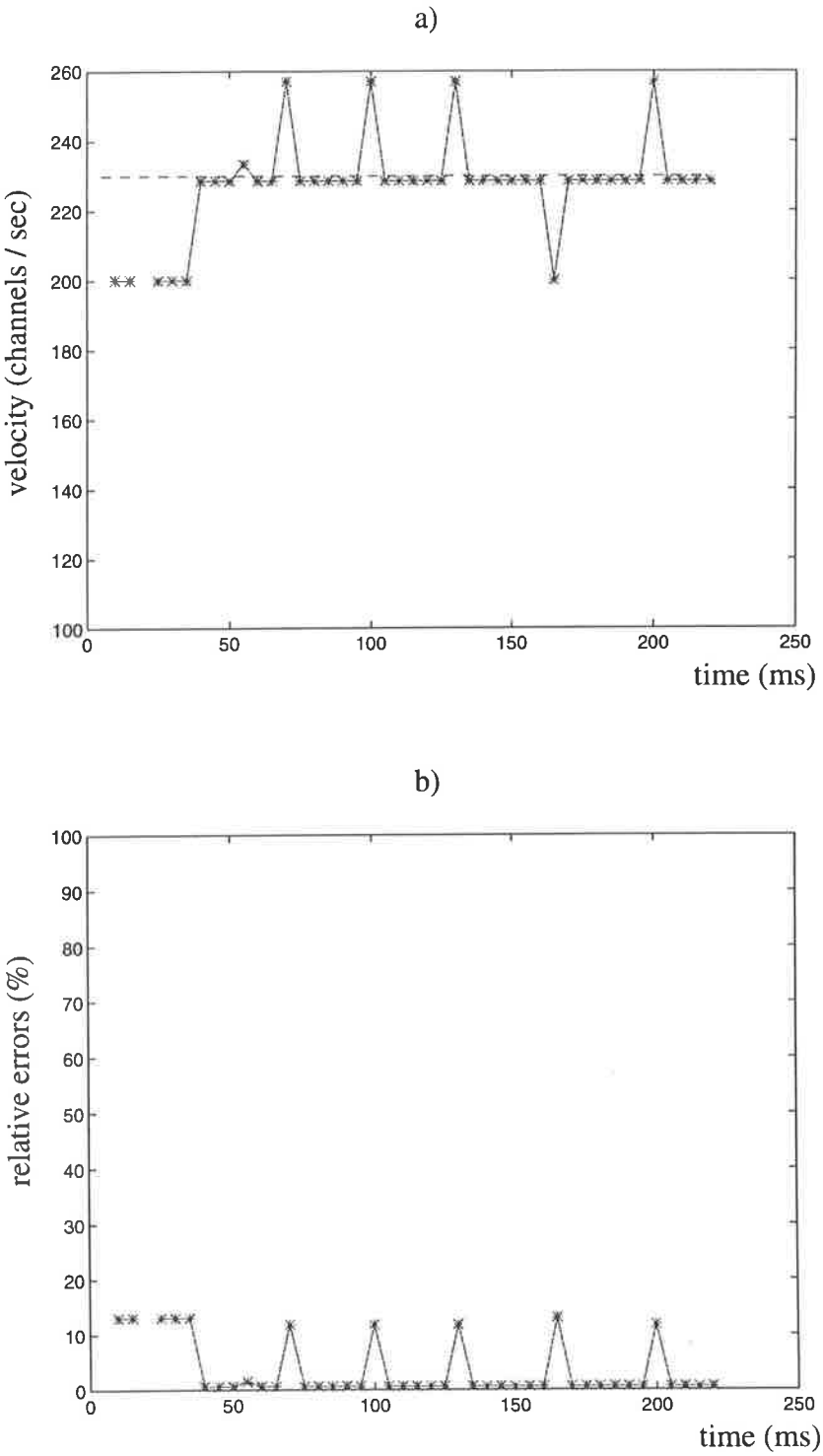


Fig. 6.8 The velocity estimation using the BT algorithm. a) The dashed line represents the true velocity of the moving object. The solid curve passing through symbols ('*') is the estimated velocity. b) The magnitude of relative errors of the estimated velocity.

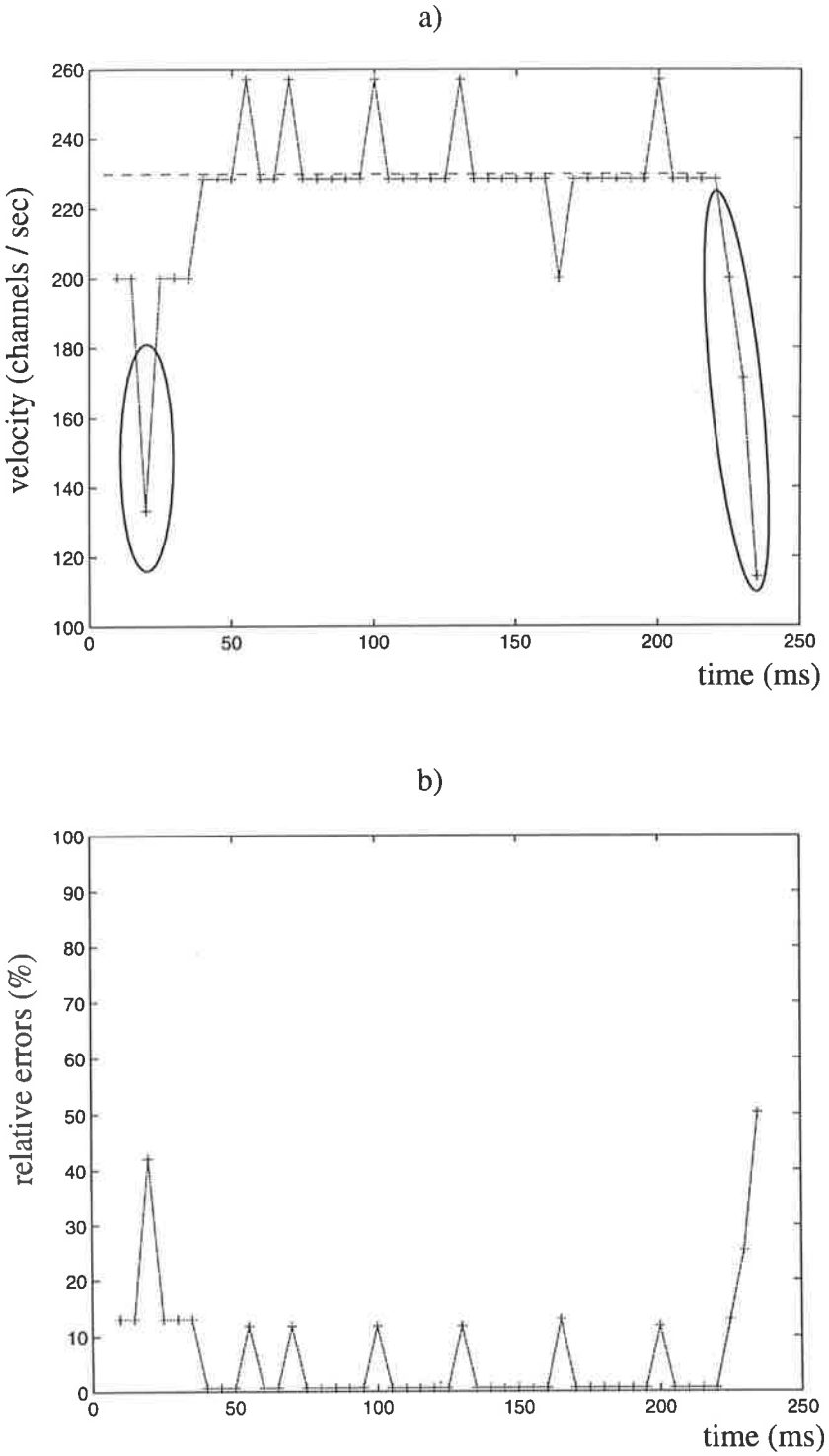


Fig. 6.9 The velocity estimation using the FT algorithm. a) The dashed line represents the true velocity of the uniformly moving object. The solid curve passing through symbols ('+') is the estimated velocity. b) The magnitude of relative errors of the estimated velocity.

estimates at the end of velocity curve as circled in Fig. 6.9a. The difference in the performance of the FT algorithm from the SST and BT algorithms can be explained by the fact that once a tracking engine is engaged in detecting a moving object and if the engine does not detect the target template for some sampling instants, it assumes that motion is slow and still reports a new velocity estimate each time. This assumption is good for detecting slow motion or missing target templates such as filling the gap. In contrast, this algorithm still reports estimates at the end of a motion trajectory where an object may stop moving, such as marked in Fig. 6.9a, or change its heading direction.

Thus, in the case of well-established motion trajectories, the performance of the BT and SST algorithms are identical. In this simulation, the algorithms can estimate the velocity of moving objects with an approximate error of 13%, which is fairly consistent with the theoretical maximum error of 12.42% due to the spatial quantization. Of course, the error may vary from time to time depending on velocities of moving objects. The FT algorithm performs almost similarly to the BT algorithm, except for some low accurate estimates due to missing target templates during the course of tracking.

6.4.2 Non-Uniform Motion

6.4.2.1 Error Analysis

In the simulation of non-uniform motion, the object non-uniformly moves from left to right and back in front of the detector. The object's motion trajectory is a sinusoidal path in the spatio-temporal domain and its corresponding velocity is then obtained as the derivative of the trajectory with respect to time. The motion trajectory is described by

$$x(t) = A\cos wt + C \tag{6.5}$$

where A and C are arbitrary constants, and w is angular frequency (radians per second). The corresponding velocity is

$$v = \frac{dx}{dt} = -Aw\sin wt. \tag{6.6}$$

Fig. 6.10 shows the true velocities of the object, plotted as a solid curve for $A = 25$ and $w = 10$ radians per second. For the purpose of comparison with simulation results, linear approximated velocities using the linear approximation are also calculated by determining two points on the object's motion trajectory described in Eq. (6.5) and a linear approximated velocity is calculated as the slope of a straight line connecting those two points. For consistency, the points are 7 sampling intervals apart (i.e., 35 milliseconds). For example as shown in Fig. 6.1, the linearly approximated velocity at point A is the slope of the segment AB. The linear approximated velocities are also plotted as a dashed curve in Fig. 6.10. The dashed

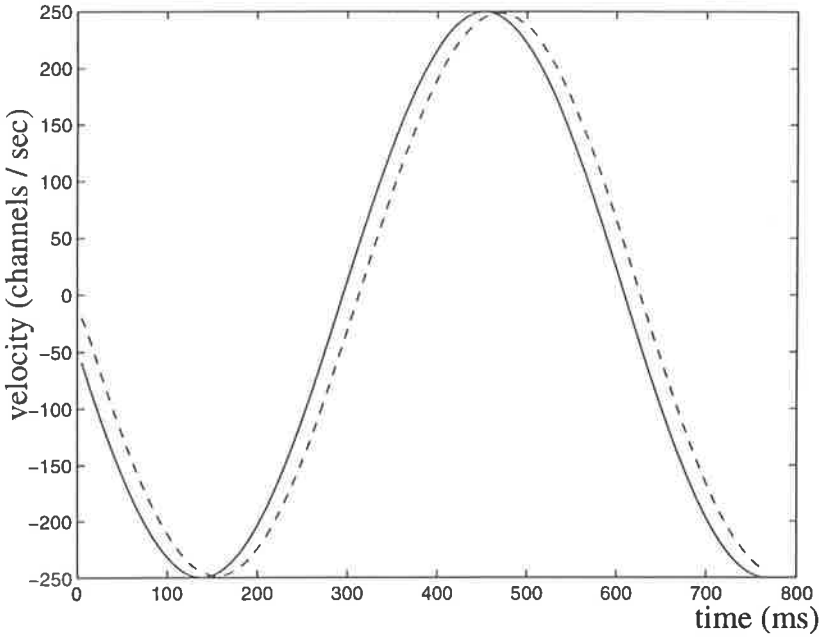


Fig. 6.10 Theoretical velocity curves. Solid curve represents velocities obtained from Eq. (6.6). Dashed curve represents velocities calculated using the linear approximation.

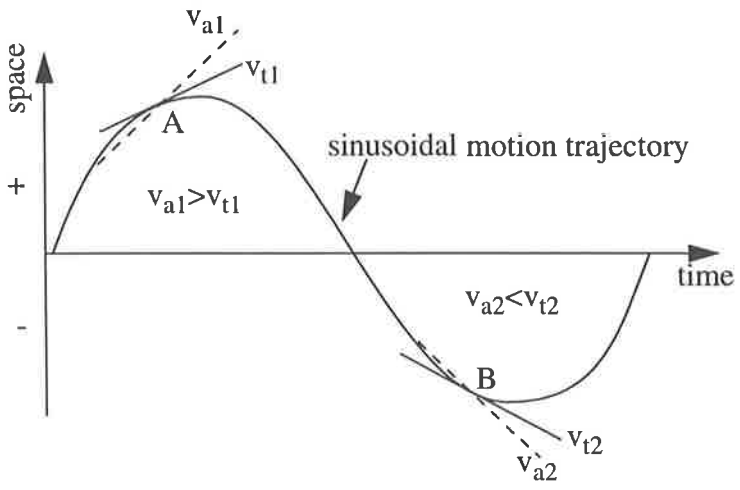


Fig. 6.11 Linear approximation of speeds.

curve is shifted to the right of the solid curve due to the linear approximation of speeds. Fig. 6.11 shows that, for a sinusoidal motion trajectory, the linear approximated velocities using the linear approximation are higher than the true velocities for positive half of a cycle, and conversely, the theoretical estimates are lower than the true velocities for negative half of the cycle. Therefore, the theoretical estimate curve is shifted to the right of the true velocity curve. Thus, in the case of non-uniform motion, in addition to the spatial quantization, the linear approximation of speeds also contributes to the accuracy of estimated velocities. The theoretical error due to the linear approximation is calculated as

$$\text{approximation error} = \frac{100 \times |\text{true velocity} - \text{approximated velocity}|}{|\text{true velocity}|} \quad \%$$

On the other hand, taking into account of the effect of the spatial quantization on uniform motion, the linear approximated velocity is used to calculate the maximum error due to the spatial quantization (see Appendix E). For a channel size of 1 degree, the maximum quantization error is given by

$$\text{maximum quantization error} = \frac{100}{\text{theoretical estimated velocity} \times 0.035} \quad \%$$

Finally, the theoretical error is computed as the sum of the approximation and maximum. That is, the theoretical error is given by

$$\text{theoretical error} = \text{approximation error} + \text{maximum quantization error} \quad \%$$

A reason for accumulating the two errors is explained as following. Let e_1 be the relative error of the linear approximated velocity v_1, e_2 be the relative error of the estimated velocity v_2, and v_0 be the true velocity, then the theoretical error e_t is the sum of e_1 and e_2 as shown below.

$$v_1 = v_0 (1 + e_1)$$
$$v_2 = v_1 (1 + e_2)$$

Hence,

$$v_2 = v_0 (1 + e_1)\ (1 + e_2)$$
$$v_2 \approx v_0 + v_0 (e_1 + e_2) \qquad \text{for small } e_1 \text{ and } e_2$$

where

$$e_t = e_1 + e_2$$

Of course, the maximum quantization and approximation errors may cancel each other in practice. However, as the absolute values of the two errors are considered in this analysis, the theoretical error would go up to 200%. The theoretical error is used for comparison with the error of the estimated velocity obtained from the tracking algorithms in the next sub-section.

6.4.2.2 Results

A part of the resultant spatio-temporal image of the DMSTs and PCTs is shown in Fig. 6.12. Similar to the case of uniform motion, movement to the right of its leading edge can be recognised by the pair of templates (A,1). Movement to the left is indicated by the pair of templates (E,5). These templates are used as the target templates in the tracking algorithms.

Fig. 6.13, Fig. 6.14, and Fig. 6.15 show the results of velocity estimation using SST, BT and FT algorithms respectively. In these figures, the true velocities of the object calculated by Eq. (6.6) are plotted as solid curves, the linear approximated velocities are plotted as dashed curves, and the estimated velocities are also plotted using symbols 'o', '*', and '+' (corresponding to SST, BT, and FT respectively). Corresponding relative errors are also plotted in these figures along with the theoretical errors (solid curves).

There are some very places (not shown these figures) where the theoretical errors have very large values (e.g, 5000%), which can be explained by the fact that at these places, the object is moving very slowly, and hence small variations in the linear approximated velocities can produce large errors. These large values are left out in Fig. 6.13b, Fig. 6.14b, and Fig. 6.15b for the purpose of clearly displaying small errors. Therefore, the theoretical error curve is discontinuous while the true velocity and the linear approximated velocity curves are continuous.

In general, the estimated velocities lie along the dashed curves rather than along the solid curves. This means that the estimated velocities are fairly consistent with the linear approximated velocities. In fact, Fig. 6.13b, Fig. 6.14b, and Fig. 6.15b show that the estimated errors are almost lower than the theoretical errors.

The SST technique recovers the object's speeds provided that staircase motion trajectories are completely determined. There are some instants when this technique cannot estimate velocities which are subsequently assumed to be zero. Consequently, some peak errors of 100% are obtained as marked with solid circles in Fig. 6.13. At these instants, motion trajectories cannot be back-tracked as one of the PCTs '1' or '5' is missing or is at an incorrect location due to the noise-sensitive nature of these templates (Chapter 5). The problem of non-existence of complete staircase motion trajectories (see Section 5.5.2) also affects the accuracy of the estimated velocities of the object. This is particularly true in the vicinity of

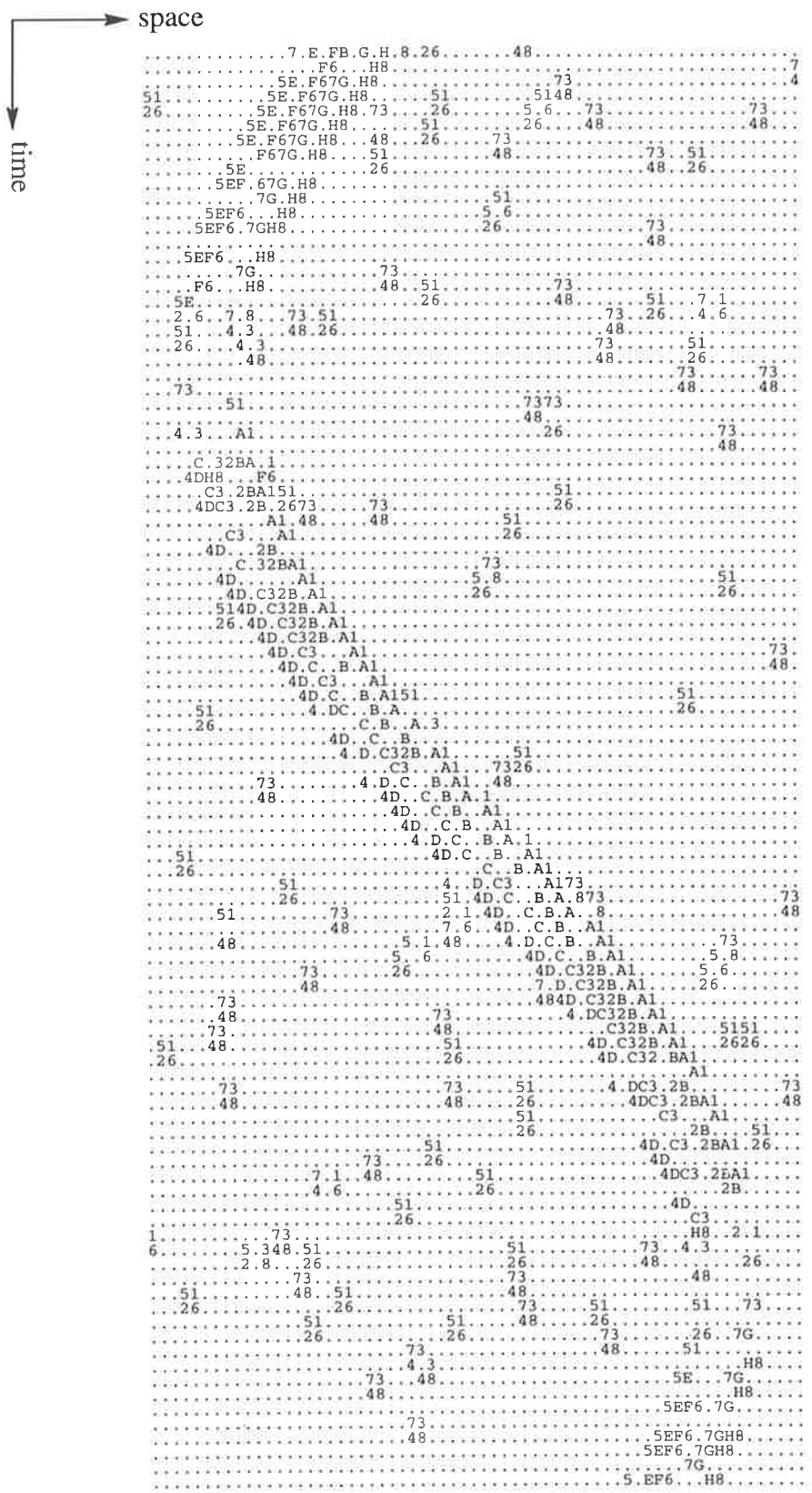


Fig. 6.12 A part of the resultant spatio-temporal image of templates created by a non-uniform movement of a object in front of the sensor. Its motion path is modelled as a sinusoidal curve.

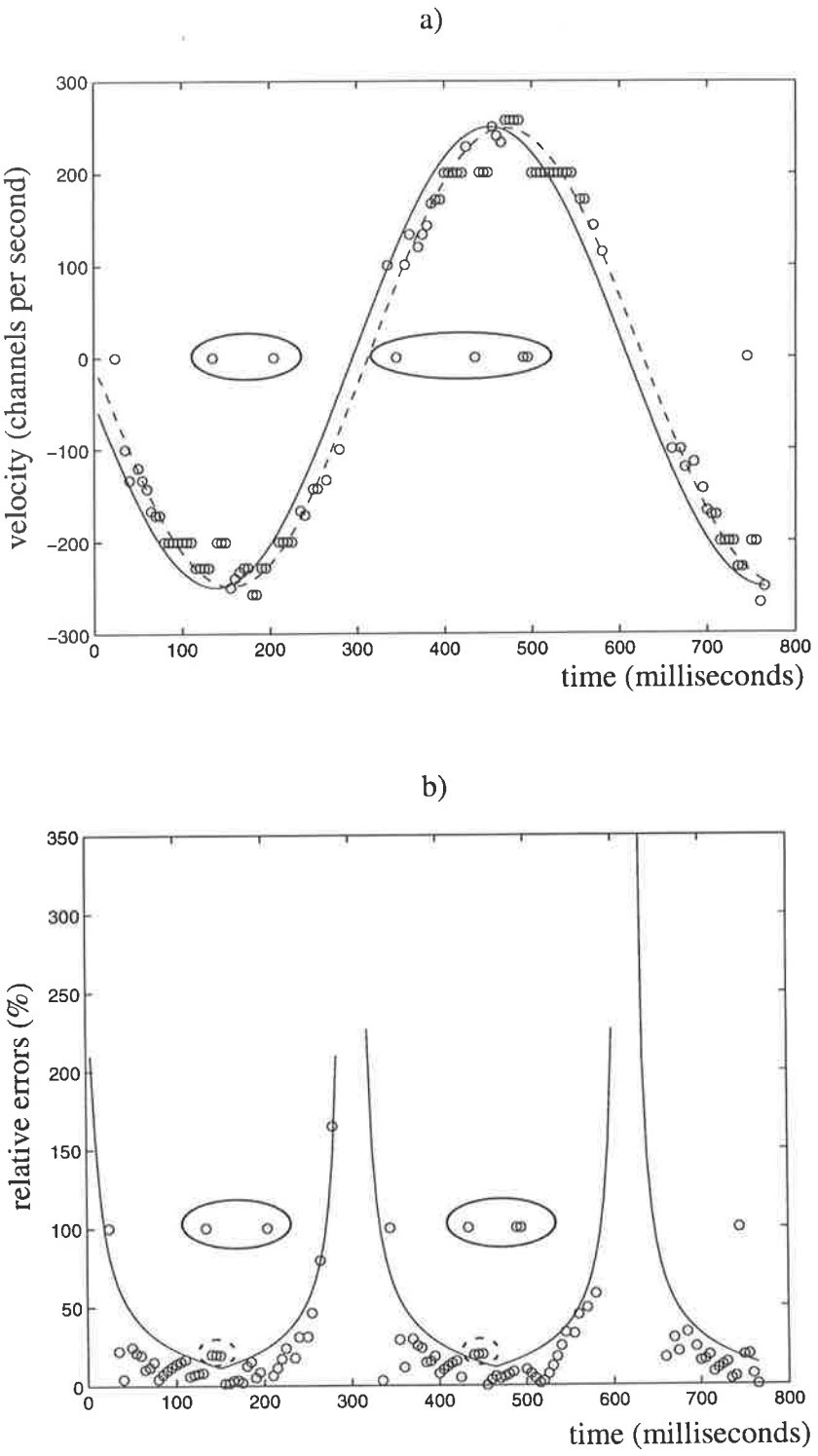


Fig. 6.13 a) Velocities. Solid curve represents the true velocity. Dashed curve represents the linearly approximated velocity. Symbols ('o') represents the estimated velocity using the SST algorithm. b) The magnitude of relative errors. Solid curve represents the theoretical error. Symbols ('o') represents the estimated error of the estimated velocity.

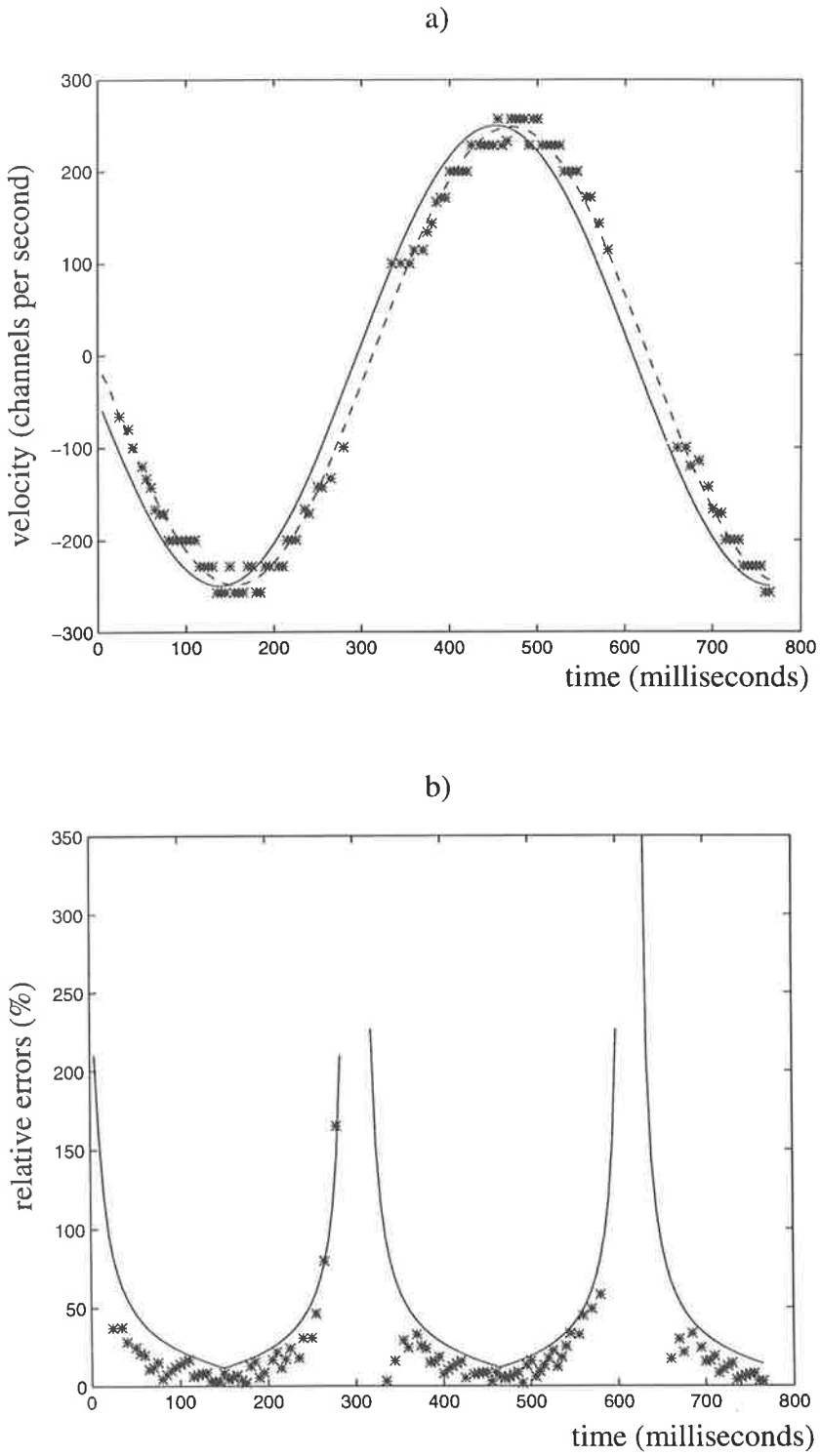


Fig. 6.14 a) Velocities. Solid curve represents the true velocity. Dashed curve represents the linearly approximated velocity. Symbols ('*') represents the estimated velocity using the BT algorithm. b) The magnitude of relative errors. Solid curve represents the theoretical error. Symbols ('*') represents the estimated error of the estimated velocity.

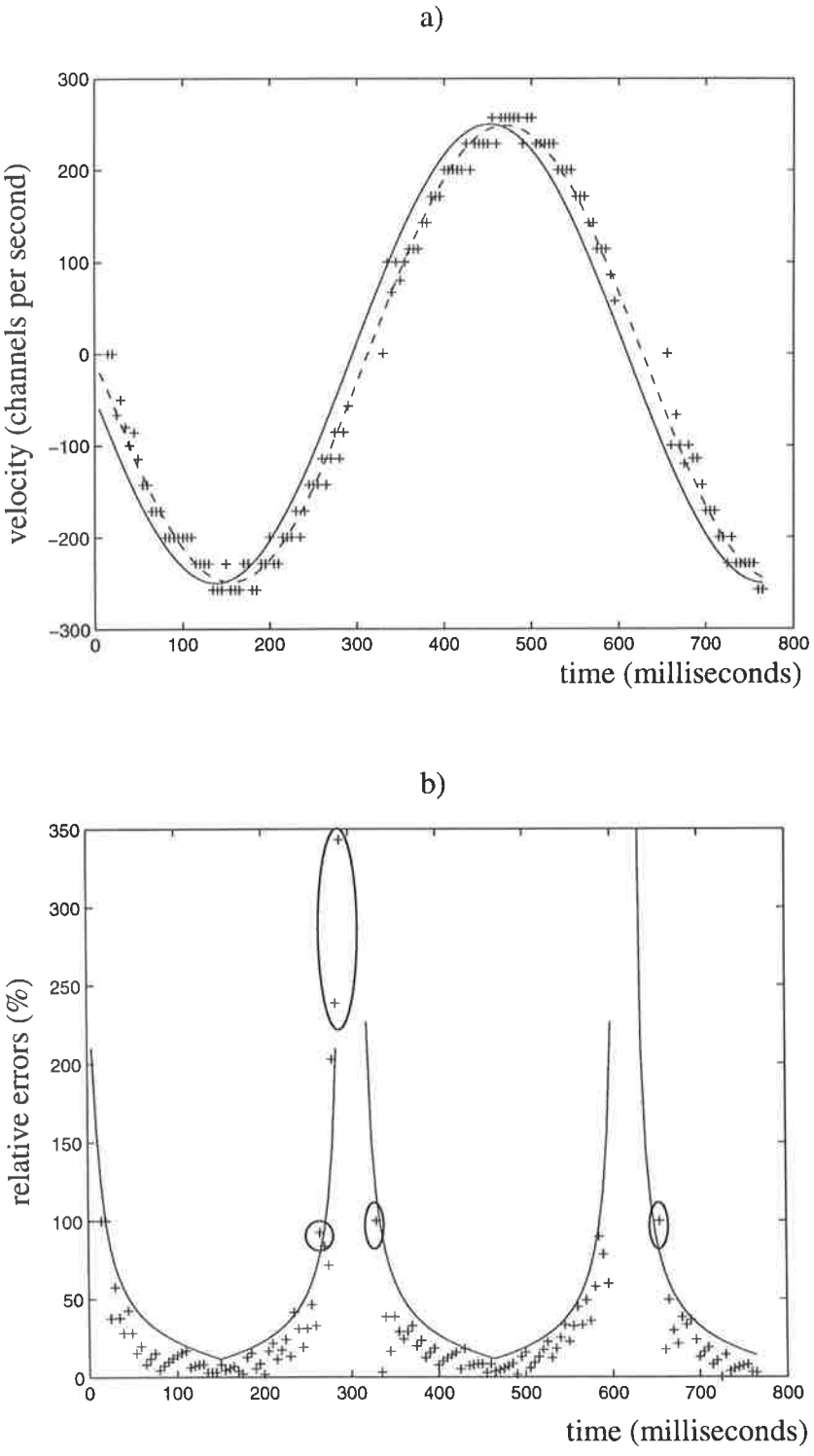


Fig. 6.15 a) Velocities. Solid curve represents the true velocity. Dashed curve represents the linearly approximated velocity. Symbols ('+') represents the estimated velocity using the FT algorithm. b) The magnitude of relative errors. Solid curve represents the theoretical error. Symbols ('+') represents the estimated error of the estimated velocity.

missing target templates where motion trajectories are usually traced for a short period of time. Hence, the corresponding estimated velocities are less accurate, up to 10% as marked with dotted circles in Fig. 6.13b, compared with those obtained from longer traces of motion trajectories.

The BT technique recovers the velocities of the object, as shown in Fig. 6.14, provided that the directionally motion-sensitive template 'A' or 'E' are detected.

Similar to the BT technique, the FT technique recovers the velocities of the object once the directionally motion-sensitive template 'A' or 'E' are detected. However, when the object is moving slowly, the directionally motion-sensitive templates do not occur for some sampling instants, the FT technique assumes that motion is slow and it still updates estimated velocities. Thus, compared to the results obtained from the BT technique, there are some extra but less accurate estimated velocities obtained from the FT as circled in Fig. 6.15b.

6.4.3 Two Moving Objects Occluding One Another

The third simulation shows two moving objects occluding one another. This simulation reflects occluding situations as described in Chapter 5. The scenario for this simulation is generated as a combination of the two previous simulations, that is, one of the objects moves constantly at a speed of 65 channels per second, while the other simultaneously moves with a time-varying velocity, whose motion trajectory is a sinusoidal path. A part of the resultant spatio-temporal image of directionally motion-sensitive templates is illustrated in Fig. 6.16.

For the purpose of demonstrating of a more realistic scenario with multi-objects moving simultaneously, only the BT algorithm is used for two following reasons. Firstly, the previous simulations show that the BT technique can produce more accurate estimated velocities compared to those obtained from the FT technique and the SST technique. Secondly, the three tracking techniques do not try to infer target templates if some of the target templates are missing. So, in the case of occluding objects where some DMSTs are missing at the occluding intersection of the objects' motion trajectories (as discussed in Section 5.6.2), it does not matter which technique is being used. In fact, the aim of the third simulation is to show that the tracking techniques may follow a wrong trajectory at the intersection if motion trajectories are represented by the same DMST, and hence they yield wrong estimated velocities, i.e., very large errors.

The DMSTs 'E' (moving to the left) and 'A' (moving to the right) are searched for in this simulation. The estimated velocities of the objects are plotted in Fig. 6.17a, and the relative errors of the estimates are shown in Fig. 6.17b. In order to provide a close view at these relative errors, separate temporal axis (1D) projections of the relative errors are plotted in Fig. 6.18a and Fig. 6.18b for the uniformly moving object and the non-uniformly moving

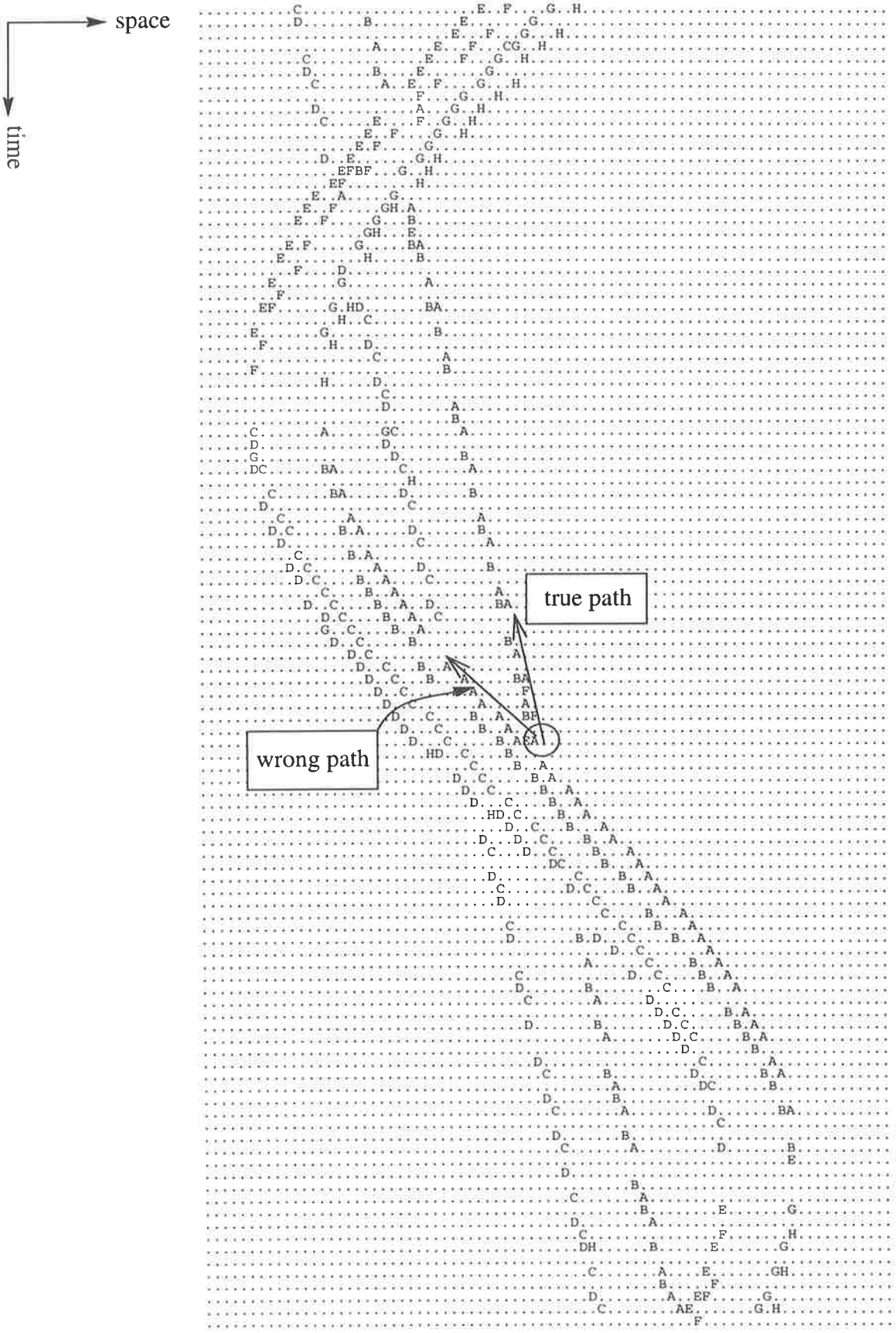


Fig. 6.16 A part of the resultant spatio-temporal image of DMSTs created by two objects moving simultaneously.

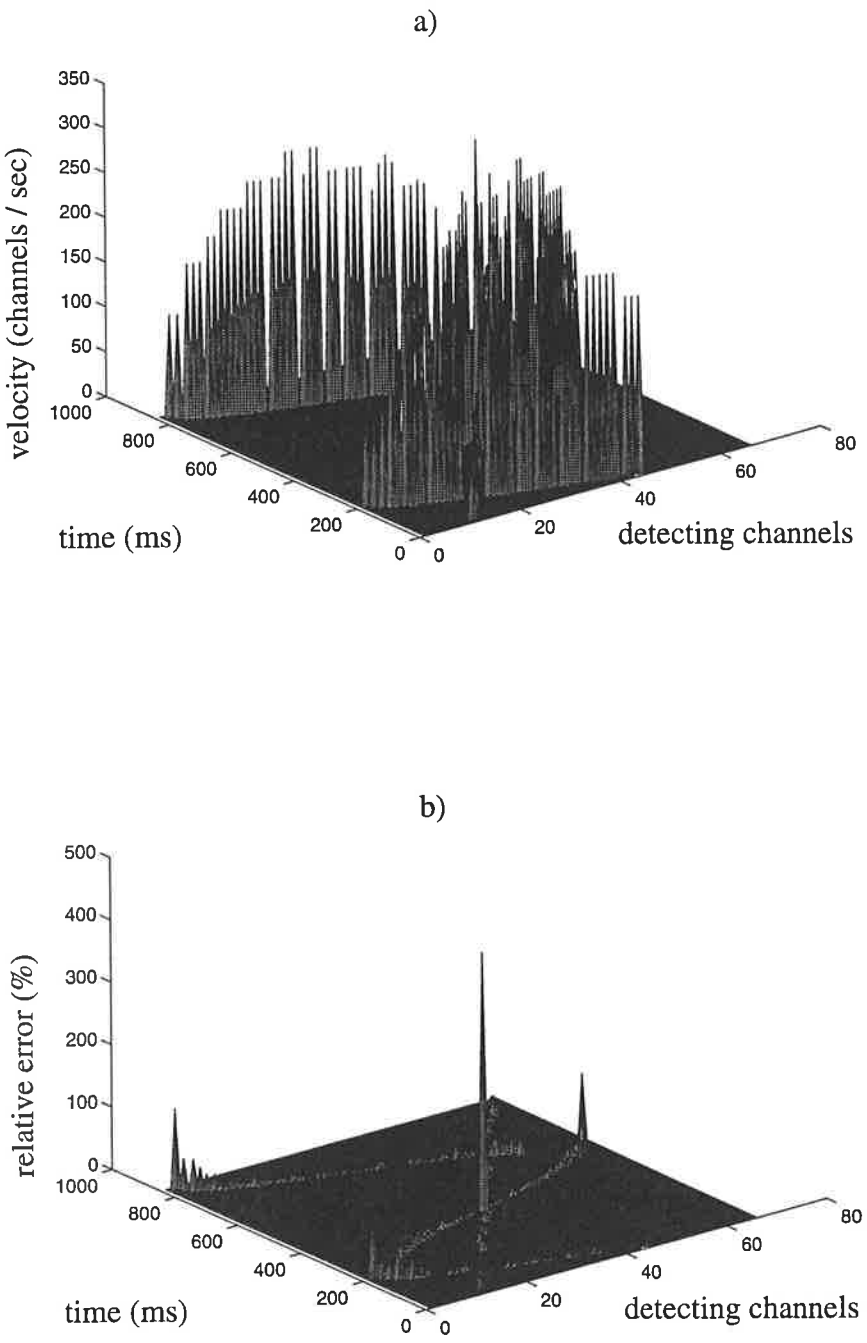


Fig. 6.17 a) Spatio-temporal profile of the estimated velocities of the two objects using backward tracking algorithm. b) The magnitude of relative errors.

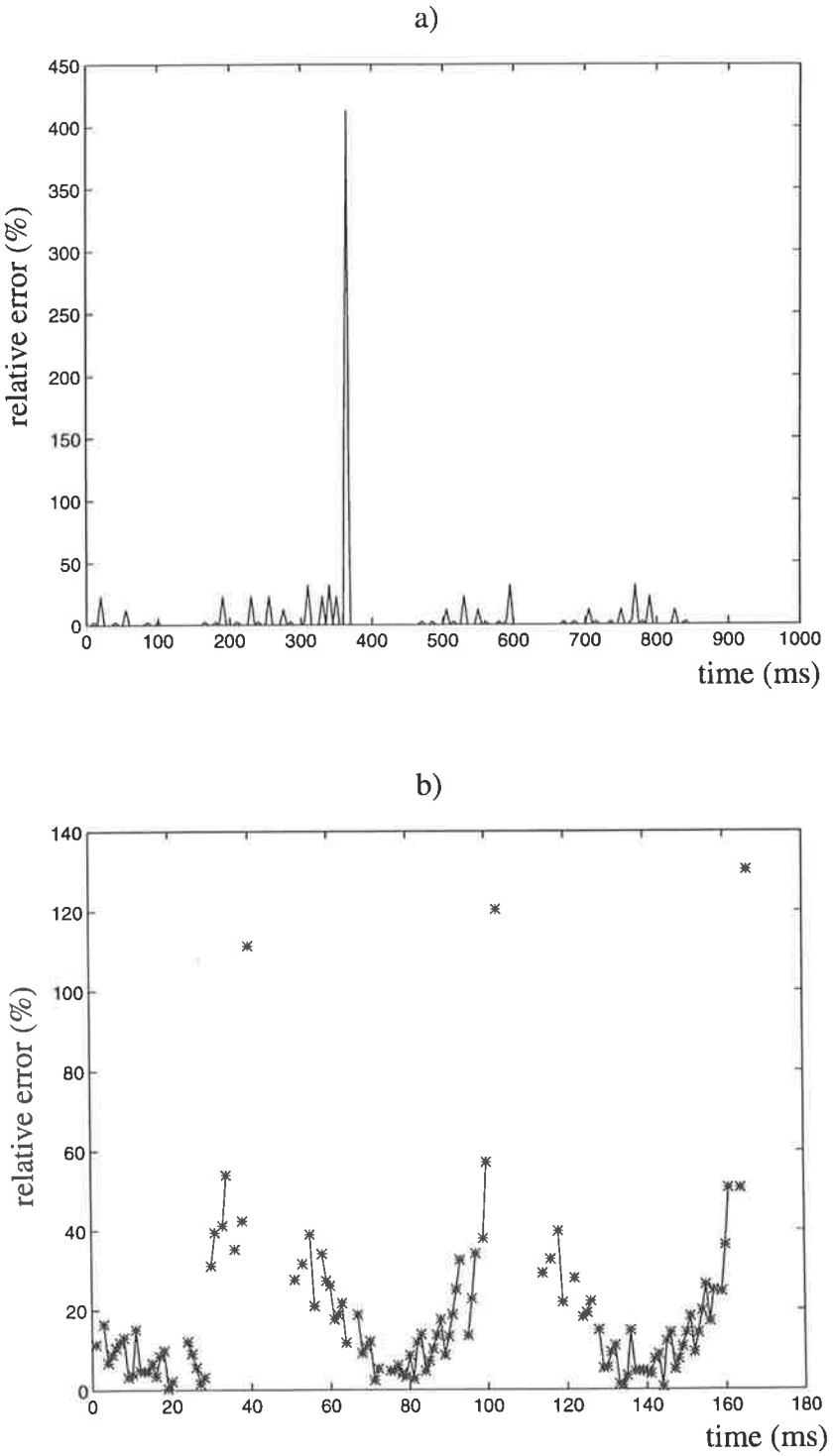


Fig. 6.18 1D projections of the magnitude of relative errors of the estimated velocity of a) Uniformly moving object, and b) Non-uniformly moving object.

object respectively.In the time-varying velocity case, similar to the results in the simulation of non-uniform motion, the errors of the estimated velocities depend upon the object's speeds, and often vary within a large range, from 5% to 60%. There are some peak errors which are again due to very slow motion.

In the constant velocity case, the errors of the estimated velocities vary in a range of 3% to 30% because the object's speed is quite low. However, there is one peak error of nearly 500% seen in the constant velocity case. This large error is due to a confusion between the motion trajectories at the occluding intersection as shown in Fig. 6.16. Instead of following the trajectory of the uniform motion from the circled starting point of the template 'A', the BT algorithm actually follows the trajectory of the non-uniform motion because the algorithm cannot distinguish different paths at the intersection (see Chapter 5). Fig. 6.17b shows the location of this peak error at the intersection of the motion trajectories.

6.5 A Comparison

While the tracking techniques and the Hough transform (see Appendix D) aim to estimate the velocities of moving objects by either interpolating or searching for target templates, there are some distinct advantages and disadvantages among them. In order to highlight the benefits from each of the algorithms, a comparison is made on the basis of three requirements: robustness, searching efficiency and strategy, and hardware architecture complexity.

6.5.1 Robustness

Robustness is defined as the ability of algorithms (or systems) to function adequately even in noisy or complex conditions. In particular, the position conjugate templates may occur at wrong locations instead of on an expected motion path, or these templates may be missing as shown in Fig. 5.5 because they are sensitive to noise (Appendix C). The role of the robustness requirement is to ensure that if such cases do arise, the algorithms should still follow the motion trajectories of moving objects and estimate their velocities.

The **Hough transform**, the **backward tracking**, and the **forward tracking** techniques search for directionally motion-sensitive templates only, while the **stair-step tracking** technique relies on finding a pair of conjugate templates: directionally motion-sensitive template, and its position conjugate template. Therefore, the SST method cannot follow any motion trajectory completely if the trajectory does not have a complete staircase due to, for example, the misbehaved response of the PCTs caused by noise. Thus, the SST method is more susceptible to noise than the other three techniques. In other words, it is less robust than the others.

6.5.2 Searching Efficiency and Strategy

The tracking algorithms and the Hough technique employ different searching strategies. The searching efficiency is defined as how fast an algorithm can search for a motion trajectory in order to estimate a velocity of interest. This is important as the searching efficiency partially contributes to the efficiency of processing time as discussed in the smart-sensing paradigm (Chapter 2). Although the searching techniques of the SST, BT, and Hough transform techniques are similar in that targets are searched for in the template data, the amount of time in which each technique completes a search is different depending upon the searching strategies.

To make a fair judgement, these algorithms should be considered within the same search space as defined in Fig. 6.2, which covers from 2 up to 18 detecting channels per 7 sampling intervals. This range of velocities defines a search space of half of a 18x8 pixel-sized rectangle roughly, where a pixel is defined as a template location.

The **Hough transform** technique attempts to estimate a velocity that corresponds to the maximum number of counted templates along 17 pre-defined velocity lines (see Appendix D), therefore, target templates are searched for in the whole search space. This implies that the technique has to perform a *complete search* for all 72 pixels in the search space.

The **stair-step tracking** algorithm establishes a staircase motion trajectory in the search space by finding pairs of conjugate templates, which indicate searching directions. Thus, it is a *deterministic* search. The worst case occurs when the maximum velocity is to be tracked, which corresponds to 18/7 channels per sampling-interval. This trajectory consists of 18 pixels on the horizontal axis and 7 pixels on the vertical axis of the search space. Thus, the worst case implies a total of 25 pixels to be visited.

The **backward tracking** algorithm searches for previously occurring target templates in the vicinity of the currently occurring one. The efficiency depends mainly on the size of the window. As the maximum possible velocity is 18/7 channels per sampling-interval, it is less than 3 channels per sampling-interval, and hence, a window of 3 channels can be used. Thus, the worst case is to completely search for the target in all 7 windows corresponding to the 7 previously sampled instants. Thus, the worst case implies a total of 21 (i.e., 7x3=21) pixels to be visited.

The **forward tracking** algorithm predicts the future-occurring target templates and searches for them in the next sampling instant. This technique does not search for target templates occurring in the past, hence there in no visit to the search space. Thus, compared to the other techniques, the forward tracking algorithm has the best searching efficiency.

However, it should be noted that the above evaluation is applied to the case where only one moving object is considered. In the case of a dense image of directionally motion-sen-

sitive templates, i.e., several objects are simultaneously detected in the visual field, the search space might be the image itself. Hence, the Hough transform, stair-step and backward tracking techniques may search for targets over the whole image.

6.5.3 Hardware Architecture Complexity

The **Hough transform** technique requires two 2D arrays implemented by memories (Hanahara et al. 1988). One stores the template data while the other accommodates the counted numbers of target templates along the 16 quantized velocity slopes. The Hough transform technique also requires another look-up table for storing transformation information. This table maps a template position into counter values.

The **stair-step** and **backward tracking** algorithms differ mainly in their searching strategies which require different controllers. However, these techniques share almost the same hardware architecture of one processor which comprises of a number of components such as address generators, matching modules (Nguyen et al 1994b). Importantly, the techniques require a memory only for storing previously occurring template data.

The **forward tracking** has a quite regular architecture as a number of identical tracking engines are designed. As each tracking engine is engaged with up to two moving objects once the objects are detected, a problem of the number of available engines less than a

Techniques	Robustness	Searching Efficiency & Strategy	Simple Hardware	Other Comments
Hough transform	Yes.	Complex, 72 visits, global.	No, at least two memory, transform table, etc.	Only partial use of motion templates as feature points
stair-step tracking	No.	Good 25 visit, deterministic.	Yes, single engine, one memory.	Fully use of motion templates as feature points, and also their direction selectivity for searching strategies
backward tracking	Yes.	Good, 21 visits.	Yes, single engine, one memory.	
forward tracking	Yes.	Best.	Yes, multi engines, no memory.	

Table 6.1 The overall efficiency of the four techniques.

number of moving objects might be raised. In such cases, a few objects might not be detected. To minimise the number of undetected moving objects, a big number of engines may be required.Furthermore, although this technique does not require a template memory, each tracking engine has a small memory in the form of a FIFO register for keeping the history displacements of a target. Thus, depending on the number of tracking engines, it may end up occupying more surface area than the other tracking techniques (Yakovleff et al.1994).

Table 6.1 summarizes the features of the four techniques discussed previously.

6.6 Conclusion

In this chapter we have presented the fundamentals of velocity estimation as the linear interpolation of lines passing through the first and last occurrences of a feature template in the spatio-temporal domain. This makes the problem of velocity estimation equivalent to that of feature detection in the spatio-temporal domain.

Three different tracking algorithms, known as the stair-step (SST), backward (BT) and forward tracking (FT), have been proposed. The SST attempts to track motion trajectories by searching for conjugate pairs of templates. However, this algorithm is not very robust since the position conjugate templates are sensitive to noise (Appendix C). The BT tries to overcome this problem by performing a search for only directionally motion-sensitive templates occurring on motion trajectories as feature points. The FT functions similarly to the BT, except that once a target template is detected the FT keeps updating estimated velocities even when the target template is not found at the next succeeding sampling instants. Thus, the FT is quite useful in the case of slow motion as the velocity of a slowly moving object is updated each sampling instant. However, the FT also reports estimated velocities for an object which has been followed but may already have stopped moving or have changed its heading direction.

The three tracking algorithms have been applied to the two quantitative simulations from which the simulation results show that the error of the estimated velocity depends strongly on the object's speed and the spatial quantization of the detecting channels. In the case of several objects moving simultaneously, the tracking techniques may follow a wrong trajectory which is near occluding intersections of the objects' trajectories, and hence they compute estimated velocities which are far away from the true velocities of the objects.

In terms of hardware implementation, the SST and FT techniques share almost the same hardware architecture including one processor for searching and tracking target templates. The FT is the most efficient technique in terms of searching efficiency. However, depending on the number of tracking engines, its hardware architecture may end up occupying more surface area than that of the BT technique. Furthermore, some moving objects might

not be detected in a dynamic environment if the number of tracking engines is less than that of moving objects.

As a compromise between the three factors: hardware simplicity, robustness to noise, and efficient processing time, the BT algorithm is the most promising technique. Thus, it is chosen to be mapped into hardware as the velocity processor of the smart micro-sensor II.

However, a dominant characteristic of the FT algorithm, which is the fastest estimation in terms of searching time, makes this algorithm more attractive in real-time applications that really need fast reactions towards the environment regardless of the accuracy of estimated velocities. Hence, it would be desirable to implement both the FT and BT algorithms in one velocity estimation processor with a view to providing more useful information to a real-time control system which may have different tasks utilising the information (Yakovleff et al. 1994; Yakovleff private communication).

PART 2: IMPLEMENTATION

Chapter 7

Smart Micro-Sensor I

7.1 Introduction

Hardware implementation is essential to demonstrate that an information processing system may be successful. This is the third level of the Marr methodology for studying and designing information processing systems such as vision systems (Chapter 1). In this level, the computational theory, data presentation, and algorithms that have been studied in the first two levels are implemented and tested in real time. Thus, following the Marr methodology, the best way to verify the performance of the template model and its associated algorithms presented in the previous chapters is to physically realize them.

Furthermore, from the point of view of the smart-sensing paradigm (see Chapter 2), the hardware design should be taken into account when considering the overall successfulness of a system. In other words, the level of smartness should also be visible in the hardware design, that is, the smartness should be encapsulated in the choice of technology, the architecture, the power consumption, the compactness of the device, and so on. These aspects are actually emphasised in the design of the smart vision systems presented in this thesis.

This chapter is devoted to the VLSI implementation and testing of the smart microsensor I (sensor I for short), which was mainly designed by Moini and Yakovleff as mentioned in Section 1.4. To a large extent, the material of this chapter has been previously reported in Yakovleff et al. (1993), Moini (1994), Yakovleff (1995) Nguyen et al. (1993a). The primary processing mechanism of this sensor is based upon the template model. The system's general requirements including the functional architecture of the system and the choice of the technology are described first in Section 7.2, followed in Section 7.3 by a detailed discussion of the system's VSLI design choices and architecture. Experimental tests and results are presented in Section 7.4, and the chapter is summarised in Section 7.5.

7.2 General Description

The design of the smart micro-sensor I was meant as a concept demonstrator for the smart sensor framework, where a simple but efficient processing mechanism should be elegantly implemented in hardware so as to yield a useful real-time system. The template model, which was discussed in Chapter 4, is particularly suited to being implemented in VLSI due to its simple architectural nature. With most standard VLSI technologies, photodetectors can be realized using some fairly well-understood characteristics of parasitic elements such as *np* junction photodiodes (Mead 1989). Furthermore, the detection of changes in intensity does not require high accuracy or precise device matching, thus simplifying the design of the front-end analog circuitry. Lastly, the template model does not need an explicit multiplicative operation such as that employed in the correlation model (see Chapter 4), and signals are thresholded early and presented in a digital format, thus simplifying subsequent processing.

7.2.1 Functional Architecture

The design is essentially aimed at carrying out three tasks: detection, interpretation, and velocity estimation. Each of these tasks is translated into a separate component of the architecture as illustrated in Fig. 7.1.

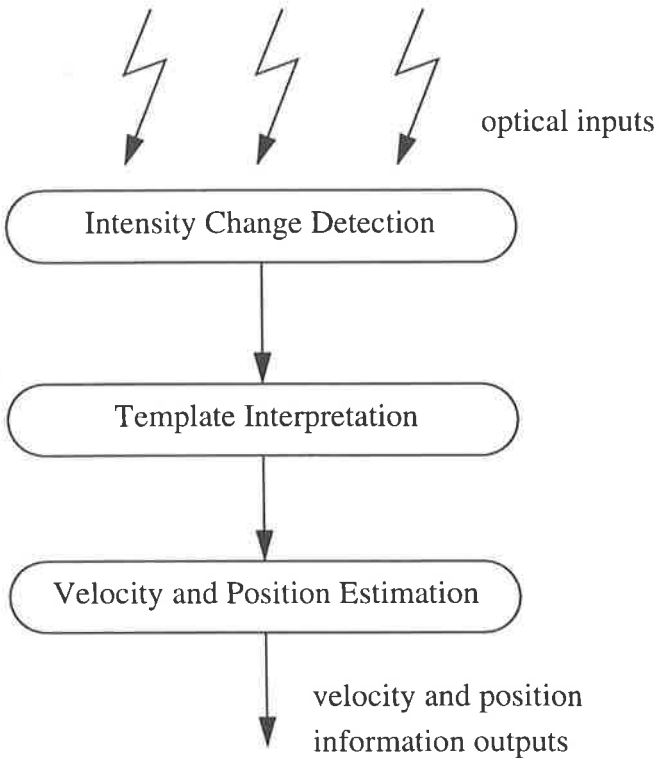


Fig. 7.1 The functional architecture of the smart micro-sensor I.

Information is first processed in many parallel channels which construct the detection stage. Each channel is responsible for monitoring changes the intensity of light focused onto its photodetector from a narrow visual field. There is no interaction between neighbouring channels in the detection stage. The outputs of this stage are presented digitally to the next stage, where templates are formed and interpreted. Templates are formed by grouping the responses of adjacent detecting channels at two consecutive sampling instances. In the third stage, the displacements of feature templates, such as directionally motion-sensitive templates (DMSTs), and their position are tracked with a view to producing velocity information.

The sampling frequency should not to be very high, otherwise changes in intensity may be too small to be detected. In addition, based upon evidence found in the insect visual system (Hausen 1982), it is suggested that the sampling frequency at the detection stage should be in the order of 100 Hz. This frequency is achievable using analog VLSI and should be adequate for most real-time applications.

7.2.2 Choice of Technology

In designing this sensor, CMOS technology is preferred for a number of reasons over other existing VLSI technologies such as Gallium Arsenide (GaAs). It is well-known that CMOS technology has lower power consumption compared with GaAs technology. Although GaAs technology is more efficient than CMOS in designing optical devices such as photodetectors, since its optical sensitivity is better than that of CMOS in the visible range (Darling et al. 1988; Abbott et al. 1991), it is extremely difficult to design reliable analog circuits using GaAs technology. In comparison, CMOS analog circuits are well behaved and have been utilised for a long time. The economic factor also made a significant contribution in selecting CMOS technology. The fabrication costs in GaAs tend to be higher than that in CMOS.

Finally, the 2 μm *double-poly double-metal* CMOS technology was chosen for the following reasons. Firstly, it was an available, relatively cheap, and well-characterised process at the time the sensor I was designed. Secondly, large capacitance (in the order of 1 pico-Farad) can be achieved by superposing the two polysilicon layers, while the second metal layer can be used to mask all circuitry except photodetectors, in an attempt to prevent the effects of spurious currents generated by exposing the chip to ambient light.

In summary, the 2 μm double-poly double-metal CMOS technology offers the following benefits:

1. Well-established behaviour in either digital or analog CMOS design.
2. Low power consumption.

3. Cheap fabrication process.
4. Large capacitance and masking layer for the chip.

7.3 VLSI Implementation

7.3.1 System Architecture

The system's functional architecture is mapped into the physical VLSI architecture as shown in Fig. 7.2. The detection stage comprises a single row of 60 detecting channels, each having a photodetector, an automatic gain control, and a differentiator. Templates are formed by grouping the thresholded outputs of adjacent channels at two consecutive sampling instants. Each template is then used as an address to a random access memory (RAM), named *template* RAM, which is initialised with a pre-defined template encoding, constituting the template interpretation stage. The outputs of this stage are stored sequentially in the intermediate result memory, named *result* RAM, while being transferred to the final stage for further processing. The result RAM can be also read between sampling instants. See Section 7.3.4 & Section 7.3.5 for more detailed descriptions.

This particular architecture of the template interpretation stage offers a number of benefits. Firstly, it provides flexibility, as the interpretation of the templates can be altered. Secondly, since only a small number of DMSTs and position conjugate templates (PCTs) are

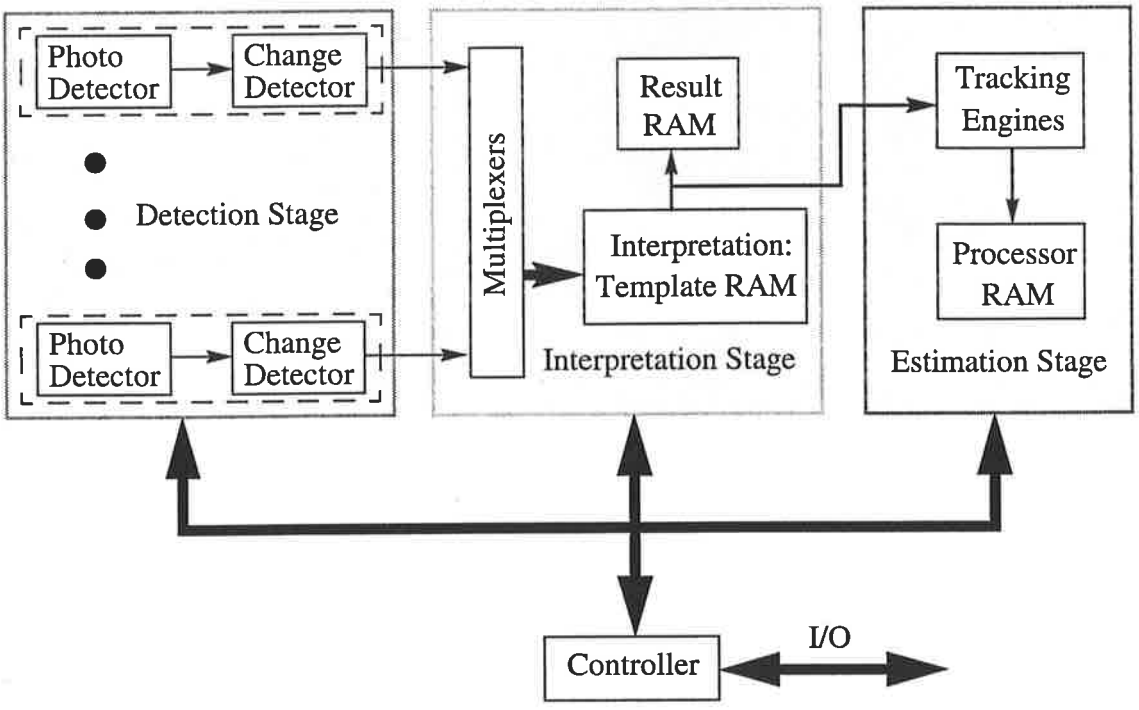


Fig. 7.2 The VLSI architecture of the smart micro-sensor I.

normally utilised for tracking motion trajectories (Chapter 6), this configuration reduces the number of bits required for a useful template representation. Since templates are encoded in 4 bits, one bit can represent the direction of motion, another edge polarity, another to differentiate between conjugate templates, and so on. Subsequently, this encoding method may minimise the computational costs of further processing stages.

The last stage of the system is the velocity and position estimation stage, where operations take place simultaneously with the storage of encoded templates into the result RAM. The estimation is carried out by a number of tracking engines.

Communication between the stages in the system is governed by controller registers. For testing purposes, the system is designed to be able to operate in different modes: normal operation, result down-load, initial template interpretation loading, and various testing modes. Communications within the system are carried out via data and address buses (Yakovleff et al. 1993; Moini 1994).

7.3.2 Timing

Timing is a very critical issue in any real-time systems. However, with advanced CMOS technology, an integrated circuit device can easily be designed to operate at much higher frequencies than the required sampling rate of 100 Hz. Therefore, processing speed is not an overriding concern in this design. This fact allows us to increase the processing frequency so that all signals from the detection stage can be processed in 10ms before the next sampling instant. This is crucial in the case of real-time applications where control systems need quick responses in order to make decisions. In this system, the operating frequencies at the interpretation and estimation stages are 1MHz and 4MHz respectively. Outputs from

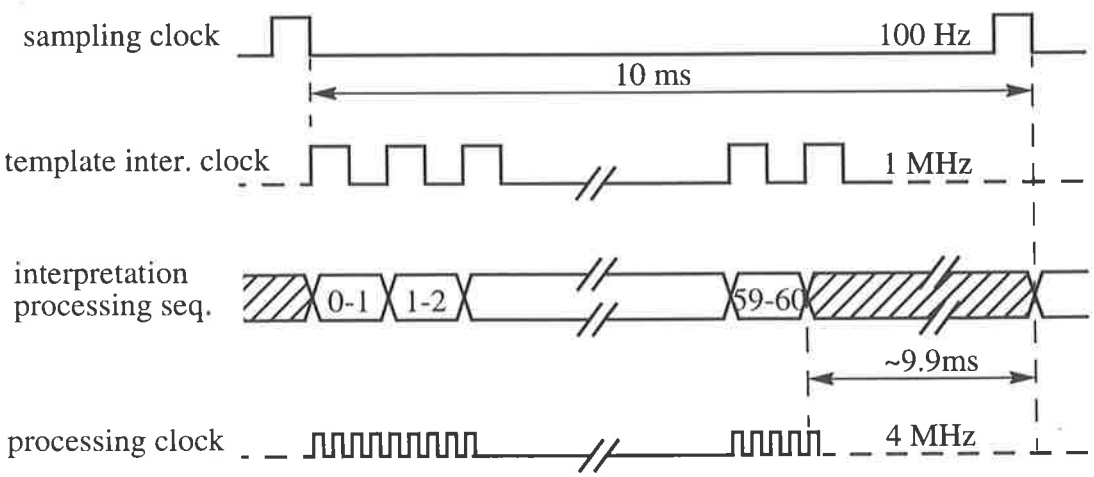


Fig. 7.3 Timing diagram for the sensor I (Adapted from Yakovleff et al. 1993).

the detection stage are multiplexed into a single interpretation processor where the template model for motion detection is actually applied. The last function of the system is to detect and track the movement of pre-determined templates by matching these templates from one sampling instant to a previous one. At a frequency of 4 MHz, the estimation stage has a time period of 9.9ms for some processing to be accomplished after templates have been interpreted. Fig. 7.3 shows the timing diagram for this system.

7.3.3 Intensity Change Detection Stage

The detection stage is the front-end stage where the interface with optics is present. It comprises a row of 61 photodetectors † which cover a visual field of 61°, hence each photodetector subtends an angle of 1° on average. A gradient index lens focuses the light onto the photodetectors. The lens is cylindrical with a diameter of 1.8mm, and has the property that the focal plane corresponds to the flat surface of the lens, and hence it is directly placed on the surface of the chip. The coupling configuration between the chip and the lens is illustrated in Fig. 7.4. In Appendix A, the optical interface of the sensor is discussed in more details.

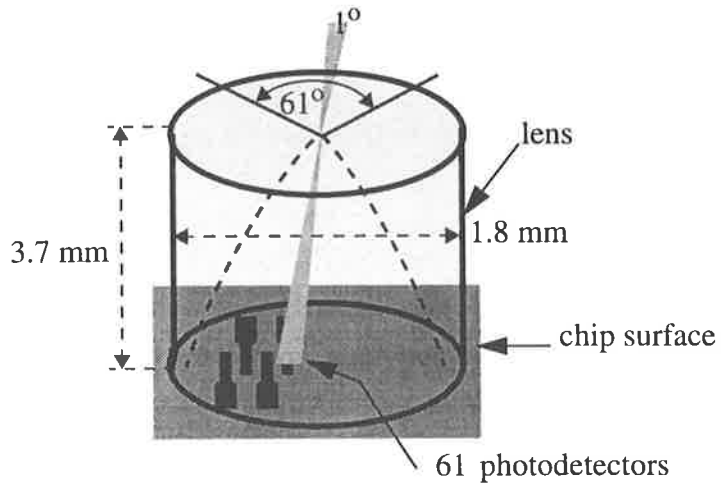


Fig. 7.4 Lens / chip configuration.

The standard CMOS process utilised for the smart sensor I can provide several parasitic diodes which can be used as photodetectors. In this system, p-well/n-epi junction photodiodes which operate in reverse-bias are selected since they can generate large photocurrents due to the wide depletion region of the well junction which has a low doping density. The photocurrents vary from hundreds of femtoamperes to tens of nanoamperes for a scene illumination of 10^{-3} to 10^{2}W/m^2. This wide dynamic range must be compressed into

†. In the detection stage 61 channels are required in order to obtain 60 templates in the interpretation stage.

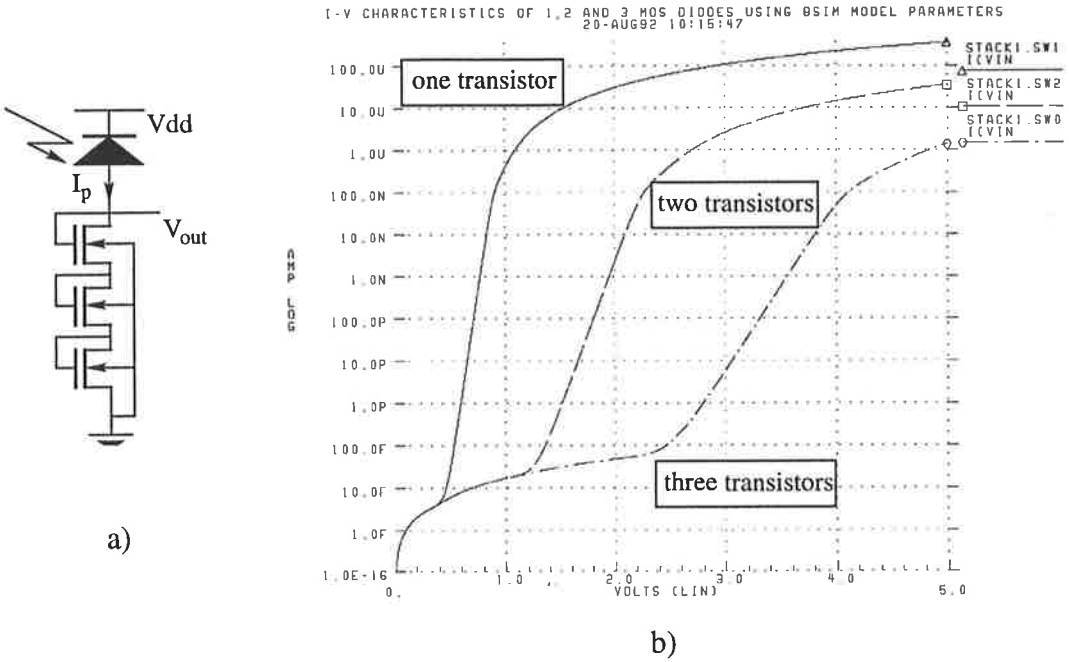


Fig. 7.5 a) The photodetector circuit consisting of a current to voltage converter and the a photodiode. b) Circuit characteristics for one, two and three transistors (Adapted from Moini 1994).

an operating range suitable to microelectronic devices. In fact, the photo current can be converted logarithmically to a voltage using a diode-connected MOS field-effect transistor in the subthreshold region (Tanner & Mead 1984; Vittoz 1985, 1994; Mead 1989a, 1990). A circuit comprising three diode-connected transistors which converts this current to a voltage is shown in Fig. 7.5.

The converter is followed by a temporal differentiator which detects changes in intensity occurring at the photodetector. The analog realisation of the differentiator can be based upon both feedback and feed-forward methods. Feedback methods are preferable as they decrease the necessity for precise matching between elements. The time constant of the differentiator is determined by the sampling frequency, that is, 10 ms approximately. The highest capacitance practically achievable in VLSI technology is in the pico-Farad range, which implies that the value of the resistor must be in the giga-Ohm range. However, since it is infeasible to design such a large value resistor with a passive element in VLSI, an active resistor with a suitable dynamic range is used instead (Moini et al. 1993, 1995; Moini 1994).

The transfer function of the feedback differentiator, as shown in Fig. 7.6, which consists of an active resistor, R, a capacitor, C, and an amplifier of gain $-k$, is described in Eq. (7.1).

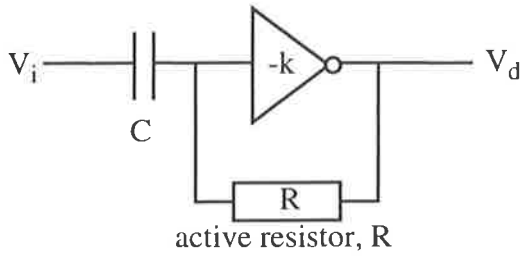


Fig. 7.6 The feedback differentiator.

$$H(s) = \frac{-kRCs}{1+k+RCs} \approx -RCs \qquad \text{if } k \ll 1 \tag{7.1}$$

It is worth noting that it is desirable to have a low-frequency bandpass filter for detecting changes in intensity so that the circuit would not amplify high-frequency noise. However, it is very difficult and expensive to design such a bandpass filter whose size would be reasonably small. Therefore, the proposed differentiator was employed instead.

The last part of the detection stage is a thresholding circuit, as shown in Fig. 7.7, which encodes the three primitive states of light intensity changes in two bits as alluded to in Section 7.3.1. However, due to the high compression of the current in the current-to-voltage converter, the output V_d of the differentiator varies within a range of only 20 to 30 millivolts around the DC value $V_{d(dc)}$. Therefore, a forward amplifier is needed to amplify V_d in order to facilitate the implementation of the thresholding circuit.

The operation of the thresholding circuit is carried out by two high-gain thresholding inverters, I_H and I_L. The characteristic of an inverter can be shifted by variation of the ratio

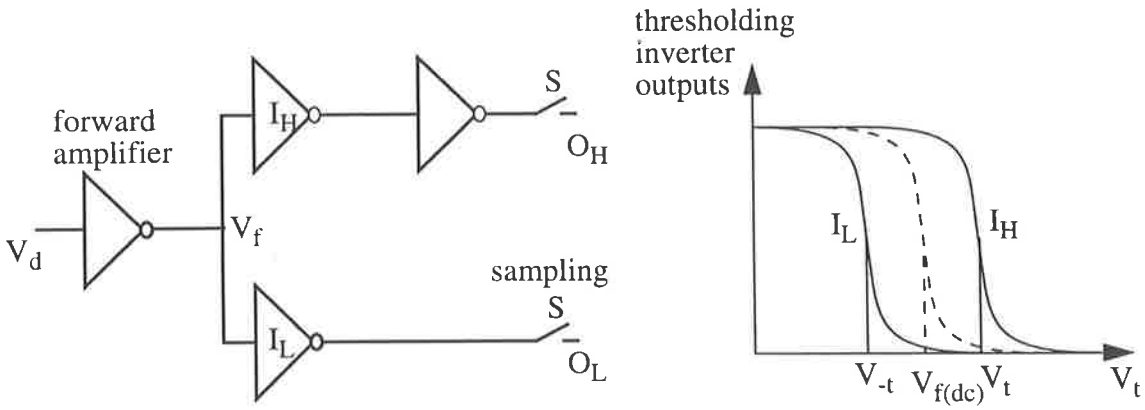


Fig. 7.7 The thresholding circuit and voltage characteristics.

of pull-up to pull-down resistances, which are in turn determined by the length to width ratios of the pull-up and pull-down transistors respectively. Therefore, the transistor sizes are chosen in such a way that the characteristic of the I_H inverter is shifted to the right from $V_{f(dc)}$ which is the DC value of the output of the forward amplifier, while the characteristic of the I_L inverter is shifted to the left from $V_{f(dc)}$, and hence provide the two selected thresholding levels, V_{-t} and V_t.

The thresholded signals are then sampled at 100Hz (10ms) and represented digitally in two bits for each primitive state as shown in Table 7.1. Thus, the thresholding circuit acts like an analog-to-digital converter as the detection circuitry itself is analog, but the intensity change detector outputs are digital.

primitive states	digital representation O_HO_L
increase	10
decrease	01
no-change	00

Table 7.1 Digital representation of the primitive states.

7.3.4 Template Interpretation Stage

The main units of this stage are two RAMs (template & result), and two addressed multiplexers, one each for odd and even channels. These multiplexers are addressed by two 5-bit-counters which are incremented alternately, thus sequentially selecting the outputs of adjacent channels. The combination of the current and stored states of adjacent channels form a template, as discussed previously. A template is used as an 8-bit address to the template RAM containing the template's useful information, which is initially loaded from outside with the template encoding. The content at a particular location addressed by the currently processed template in the template RAM is then transferred to, and stored in, the result RAM whose addresses are real physical positions of the detecting channels. The architecture of this stage is illustrated in Fig. 7.8.

7.3.5 Velocity and Position Estimation Stage

The velocity and position estimation stage detects the occurrences and relative movements of pre-defined target templates, e.g., DMSTs as shown in Fig. 4.8, and consists of six identical tracking engines and a RAM, which stores outputs evaluated by the engines. Each engine is responsible for detecting the occurrences of two particular DMSTs in a small region of the visual field (or tracking region). Thus, the visual field is effectively partitioned

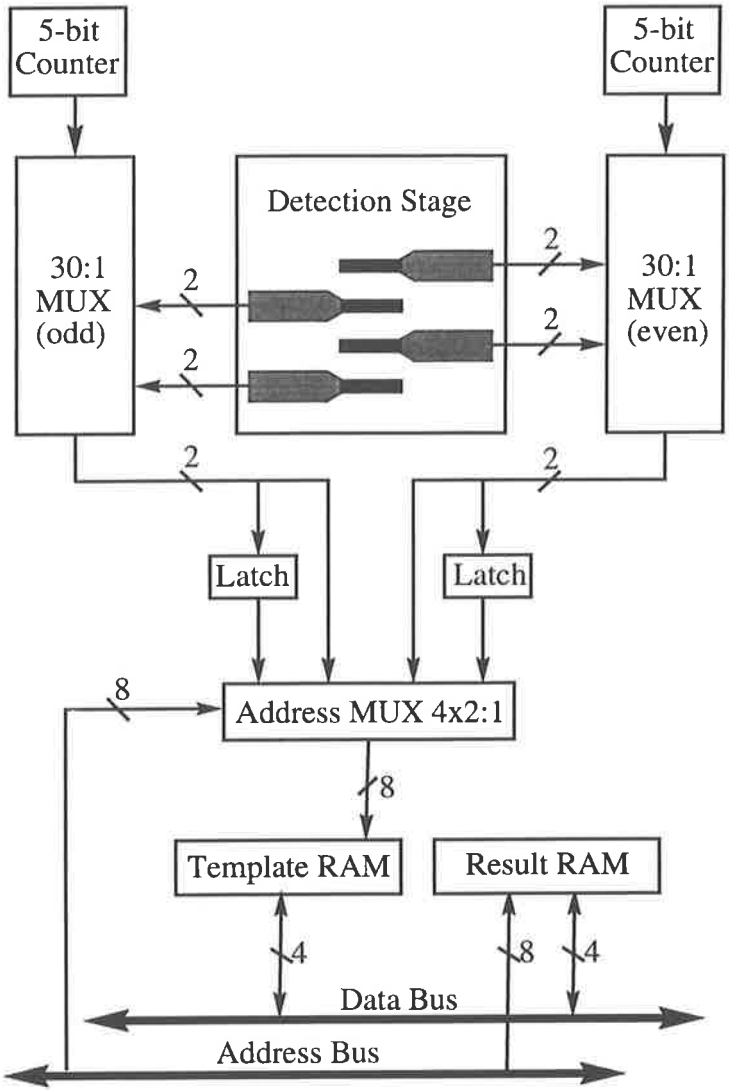


Fig. 7.8 Architecture of the template interpretation stage (Adapted from Yakovleff 1995).

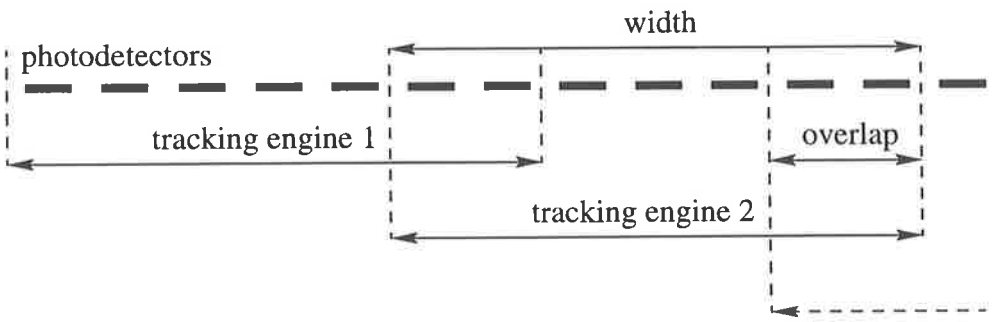


Fig. 7.9 Sub-dividing strategy of the visual field for tracking engines.

into up to 6 regions that may overlap as shown in Fig. 7.9. The width and the amount of overlap are system parameters which are set during the initialisation phase. The reasons for partitioning the visual field are mainly practical and also reflect biological findings (Yakovleff 1995). DMSTs are only associated within a given partition of the visual field, thereby minimising the effects of the correspondence problem, which may occur when there are several moving objects detected simultaneously.

Each engine comprises a couple of registers, a comparator, and a subtractor. For tracking a particular feature template, the registers store the current address input, the previously matched location, and the feature template. While a template is being stored into the result RAM, the template is also directed to the tracking engine whose tracking region accommodates the position of the template. The template is then compared with the feature template, and in case of a match, the current and previous locations are subtracted to determine the displacement of that feature template. Hence, the speed of a moving object can be deduced by taking into account the sampling frequency. The current position is then stored at the previous location register for future use. Fig. 7.10 depicts the block diagram of the tracking engine.

The current implementation of the processor suffers from three main disadvantages. Firstly, the tracking engines do not handle the situation where an object moves across the boundary between two adjacent tracking regions. Secondly, the processor tries to estimate an instantaneous value rather than an average value, and hence the velocity estimation is hardly accurate. Thirdly, the tracking engines do not use the information on the direction of motion provided by the directionally motion-sensitive templates. These disadvantages are mainly a result of inadequate information available at the time the chip was being designed.

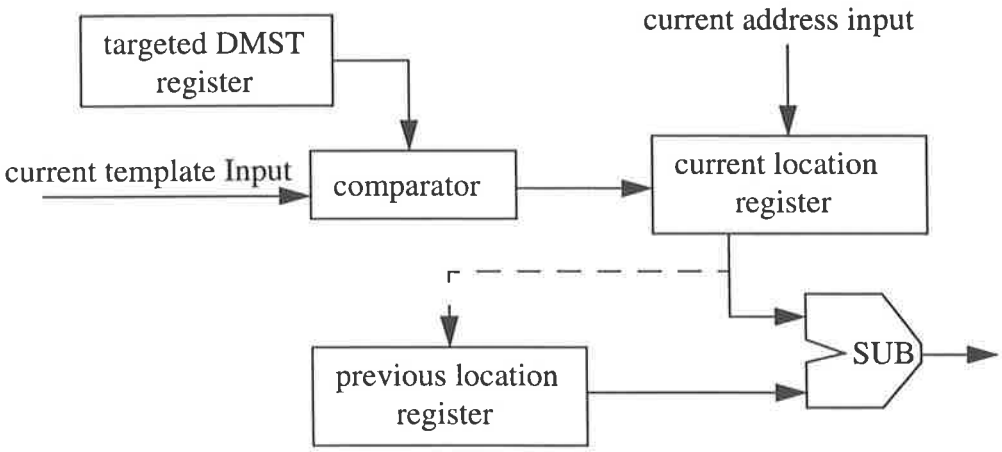


Fig. 7.10 A schematic diagram of the tracking engine.

7.4 Testing

7.4.1 Testing Procedure

The test was set up by interfacing the sensor with an IBM compatible 486DX PC via a printed circuit board (PCB) as shown in Fig. 7.11, which contains adjustable current generators for biasing the chip's analog circuits, a digital clock generator, an asynchronous reset, and a buffer for the parallel input-output port (I/O) of the PC. A gradient index lens is mounted on top of the chip's surface.

The chip is entirely under software control, i.e., control of the sampling frequency of the detection stage, as well as permitting access to the chip's internal memories and control unit. The software also allows the chip to be tested in different modes, simply by entering a single command. However, the main test of the chip is to detect motion in real time, where the template data from the result RAM are read out and displayed on the computer's visual display terminal.

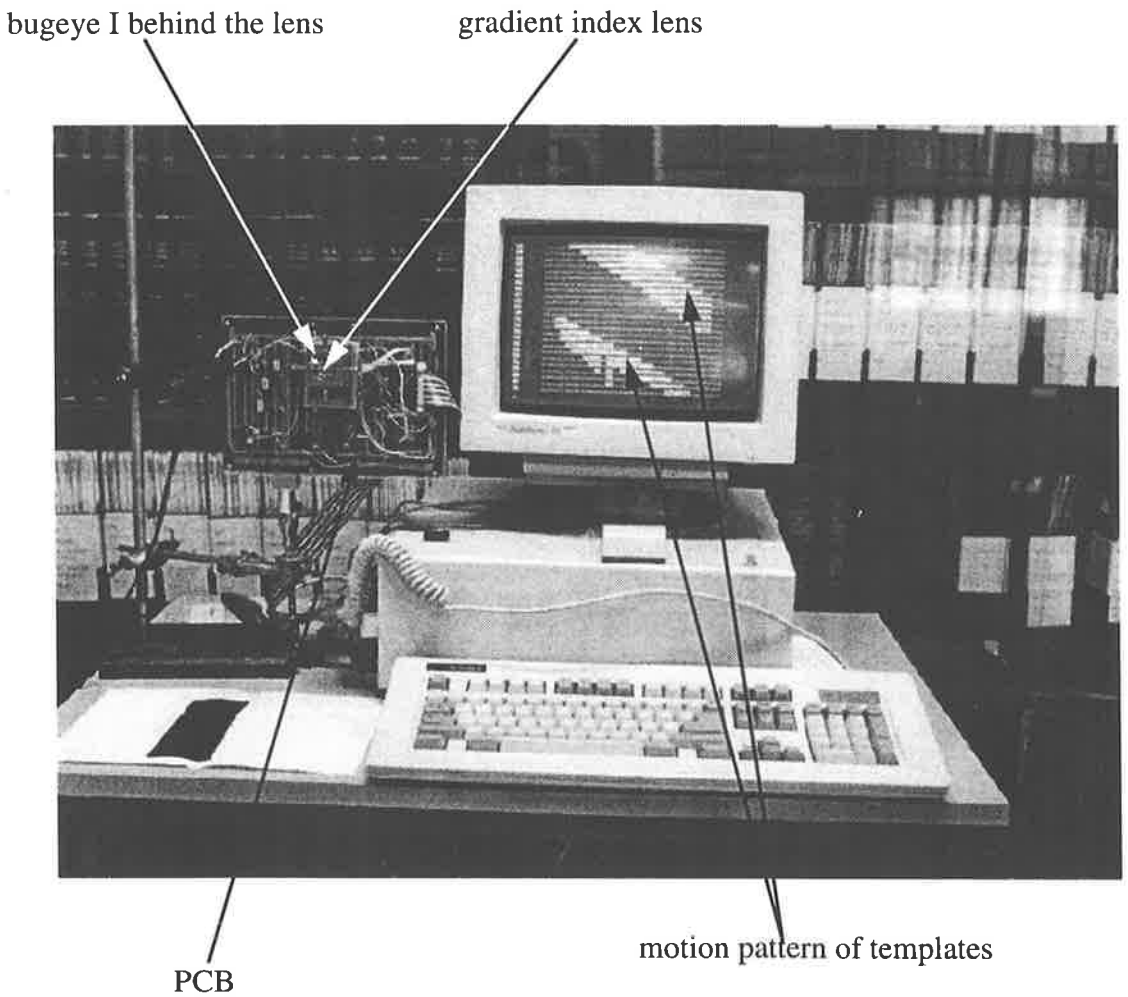


Fig. 7.11 Testing set up.

7.4.2 Interpretation of Template Formulation

The experiments were carried out under various conditions. Fig. 7.12 shows 15 templates that occurred frequently (i.e., templates are encoded by from 1 to F). Others not shown occurred very rarely and are encoded by 0. The templates are encoded as a hexadecimal number from 0 to F because the template RAM data width is four bits. Note that this encoding of these templates are different to those defined in Chapter 4 & 5.

Preliminary results indicate that the no-change template cannot be obtained reliably. Thus, instead of the template (– –/– –) for a stable background, the template (↑↑/↑↑) occurs unexpectedly pointing to a problem with the analog circuitry. Based upon experimental evidence, it is suspected that the problem is due to circuit mismatches occurring with the active resistor within the differentiator circuitry (Moini private communication). This problem can be explained by looking at the voltage characteristic of the detecting circuit consisting of the differentiator and the forward amplifier as shown in Fig. 7.13.

Recall from Fig. 7.7 that the DC value of the output of the detecting circuit $V_{f(dc)}$ would lie between the two thresholding levels V_t and V_{-t}. Hence, the output V_f around $V_{f(dc)}$, corresponding to an intensity-stable background, would yield the no-change state as shown in Fig. 7.14a. However, the circuit mismatches, caused by the fabrication process, may have changed the detecting circuit characteristics, and may consequently have lifted

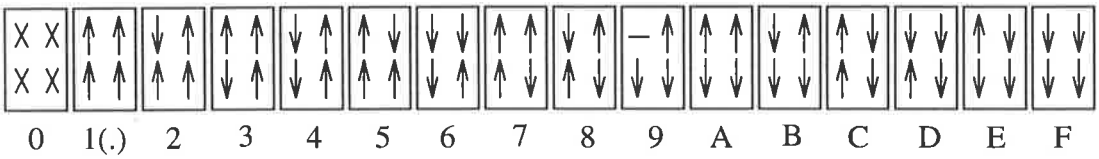


Fig. 7.12 Templates which occur most frequently in real-time experiments are arbitrarily encoded from '1' to 'F'. '0' encodes all templates which rarely occurred. Template encoding '1', which is displayed by 'dot', is obtained when the background is stable and no motion is present.

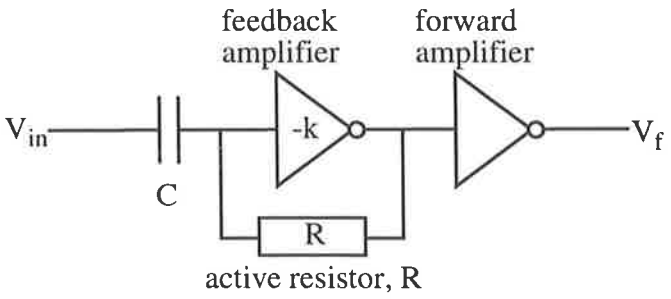


Fig. 7.13 The detecting circuit.

$V_{f(dc)}$ above the high thresholding level V_t as illustrated in Fig. 7.14b. Therefore, the increase state is almost permanently resulted in for the intensity-stable background. Thus, this problem limits the ability of sensor I to detect intensity increases due to motion of bright-to-dark edges.

However, if the increase state (↑) in the templates of Fig. 7.12 is replaced by the no-change state (–), the expected templates occur as shown in Fig. 7.15, at least in the case of a moving edge presenting an intensity decrease. Regardless of the mismatch problem, the results confirm regular occurrences of some common templates which are induced by motion, that is, directionally motion-sensitive, position conjugate templates, and so forth (Chapter 5). The templates 'B' and '3' constitute a conjugate pair indicating motion to the right, while the conjugate pair of 'E' and '7 indicates motion to the left. Templates '2', '3', '5', and '7' are the position conjugate templates associated with the directionally motion-sensitive templates '6', 'B', 'D', and 'E' respectively. Templates '4' and 'C' are spatial change templates, while '9' and 'A' are temporal change templates.

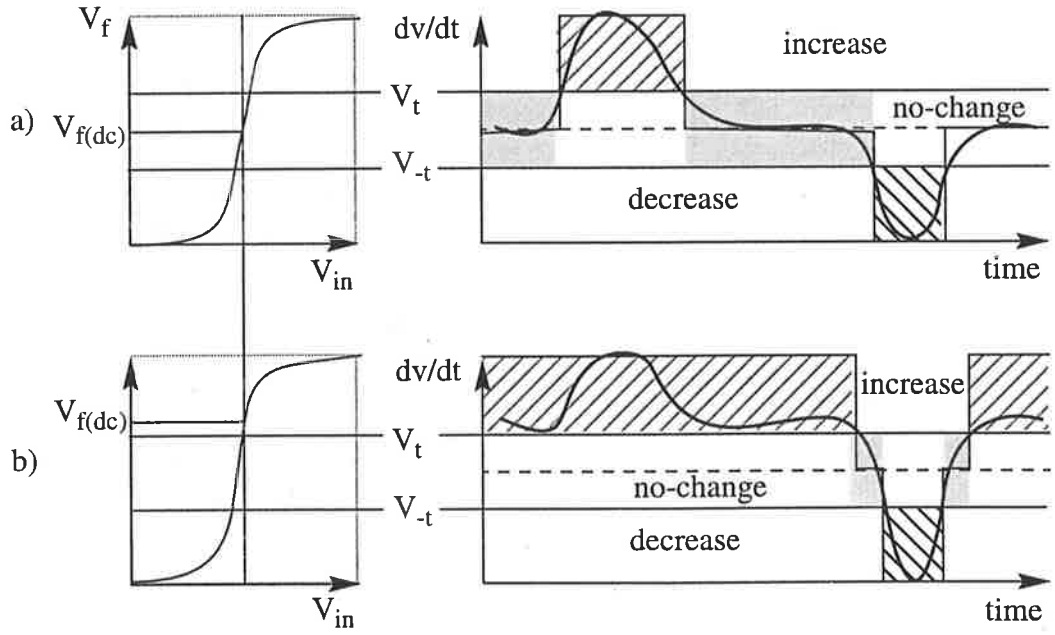


Fig. 7.14 Schematic of the circuit mismatch effect on the voltage characteristic of the detecting circuit and the output voltage of the circuit. a) Normal (theoretically interpreted) characteristic and the no-change output lies between the two thresholding levels V_t & V_{-t}, and b) changed (practically interpreted) characteristic and the no-change output lies above the positive thresholding level.

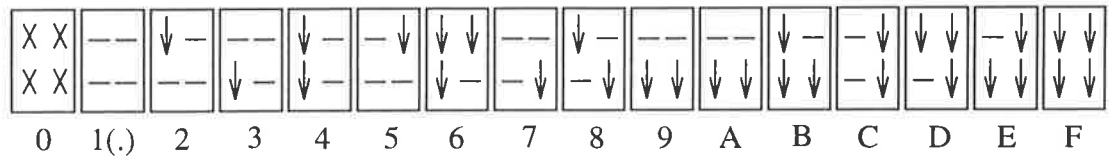


Fig. 7.15 Templates obtained from Fig. 7.12 by replacing the intensity increase state by the no-change state.

7.4.3 Real-Time Experiments

The tests were carried out under normal daylight conditions inside a laboratory with a white wall as the background at a distance of 6 meters. Fig. 7.16 - Fig. 7.19 show the outputs of the sensor detecting movements in different situations. These figures are in the spatio-temporal domain, in which the horizontal direction represents space, and the vertically downward direction represents increasing time. Fig. 7.16 corresponds to no motion, represented by 'dots'. Fig. 7.17 corresponds to motion of the dark handle of a small screw driver moving from left to right and back, about 25 centimetres away from the sensor. It is apparent that movements to the left or right are distinguished by four conjugate pairs of templates as shown in Table 7.2. These pairs sets up staircase motion trajectories (see Section 5.5.2). For instance, the pair (E,7) establishes a staircase, as shown in Fig. 7.17, on which there is one time the position conjugate template '7' does not occur on one of the concave corners as circled in the figure. Fig. 7.18 depicts the sensor's response to the motion of a person wearing a dark shirt and walking across the sensor's visual field at a distance of 3 to 4 meters. Motion at distances of up to 7 to 8 metres can be detected under ideal lighting conditions provided that there are significant differences in contrast between objects and the background. Finally, Fig. 7.19 shows the sensor's outputs to the motion of the screw driver and the person moving at the same time, and is a combination of the two responses from Fig. 7.17 and Fig. 7.18. It is apparent that the motion trajectories of the objects are not explicitly estab-

Motion to the left		Motion to the right	
motion-sensitive	position conjugate	motion sensitive	position conjugate
E	7	B	3
6	2	D	5

Table 7.2 Different pairs of conjugate templates indicating two directions of motion.

lished using conjugate pairs at the intersection of the two patterns, which is consistent with the simulation results (Section 5.6.2 & Section 5.7). However, the sensor is able to detect a few objects moving simultaneously. Note that the spatial templates 'C' and '4' appear on vertical segments of the motion trajectories, while the temporal template 'A' occurs on horizontal segments.

It is also worth noting that when experiments were carried out under fluorescent light, the 100 Hz frequency component of the light power is also picked up by the sensor, which adversely affects template patterns. This phenomenon is understandable because the luminance of the object is proportional to the illuminance of the light source (Jähne 1993).

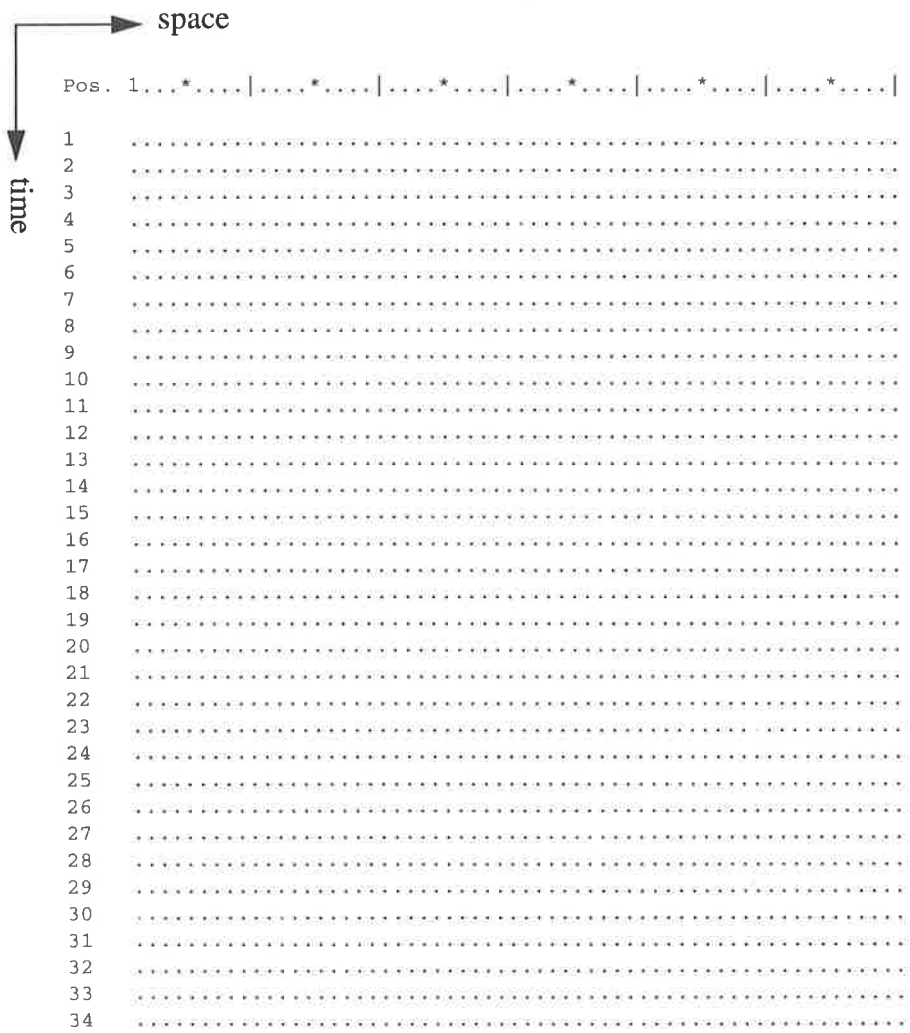


Fig. 7.16 No motion is detected.

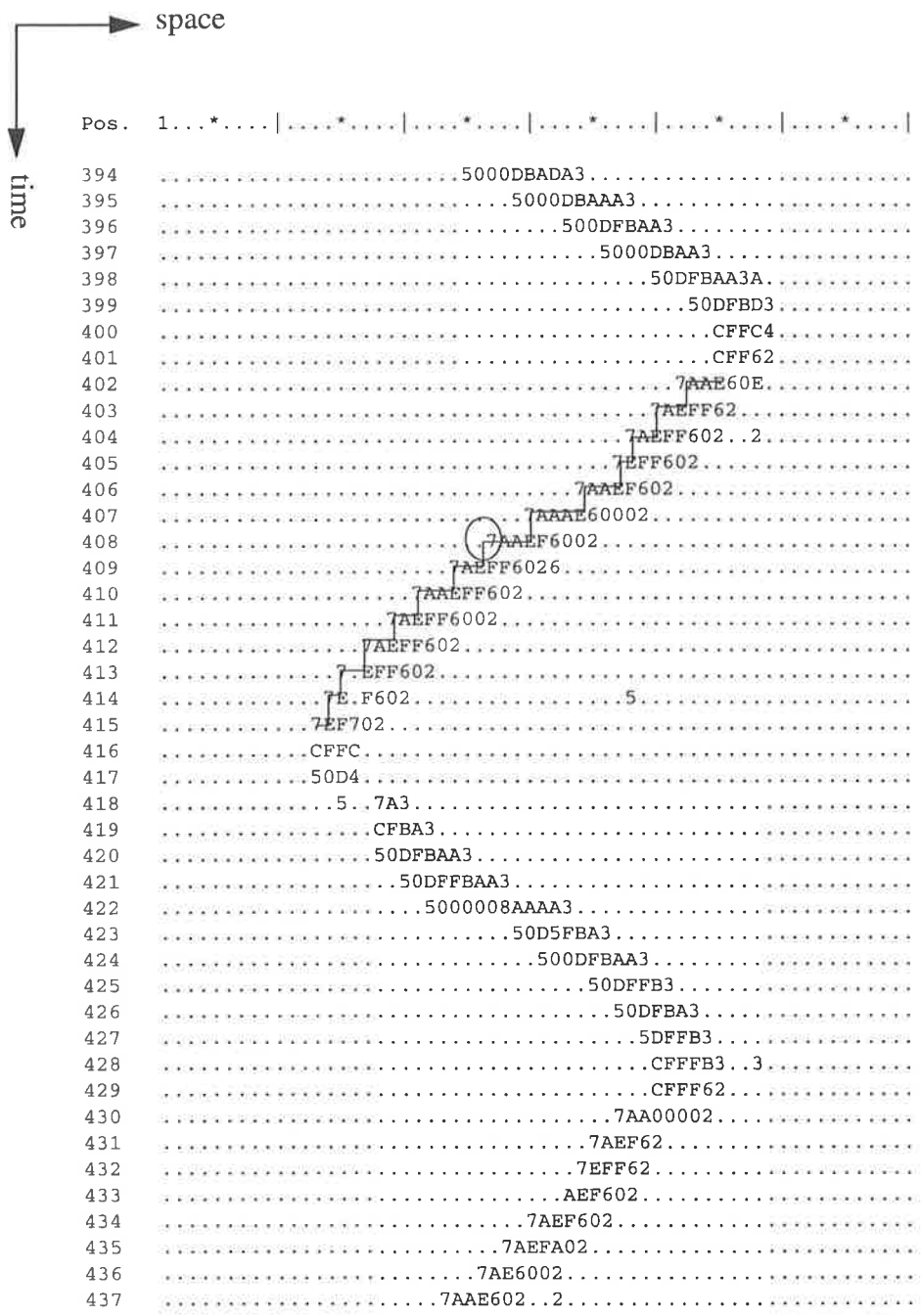


Fig. 7.17 Motion pattern of a small object moving from left to right and then right to left at a distance of roughly 25cm from the sensor.

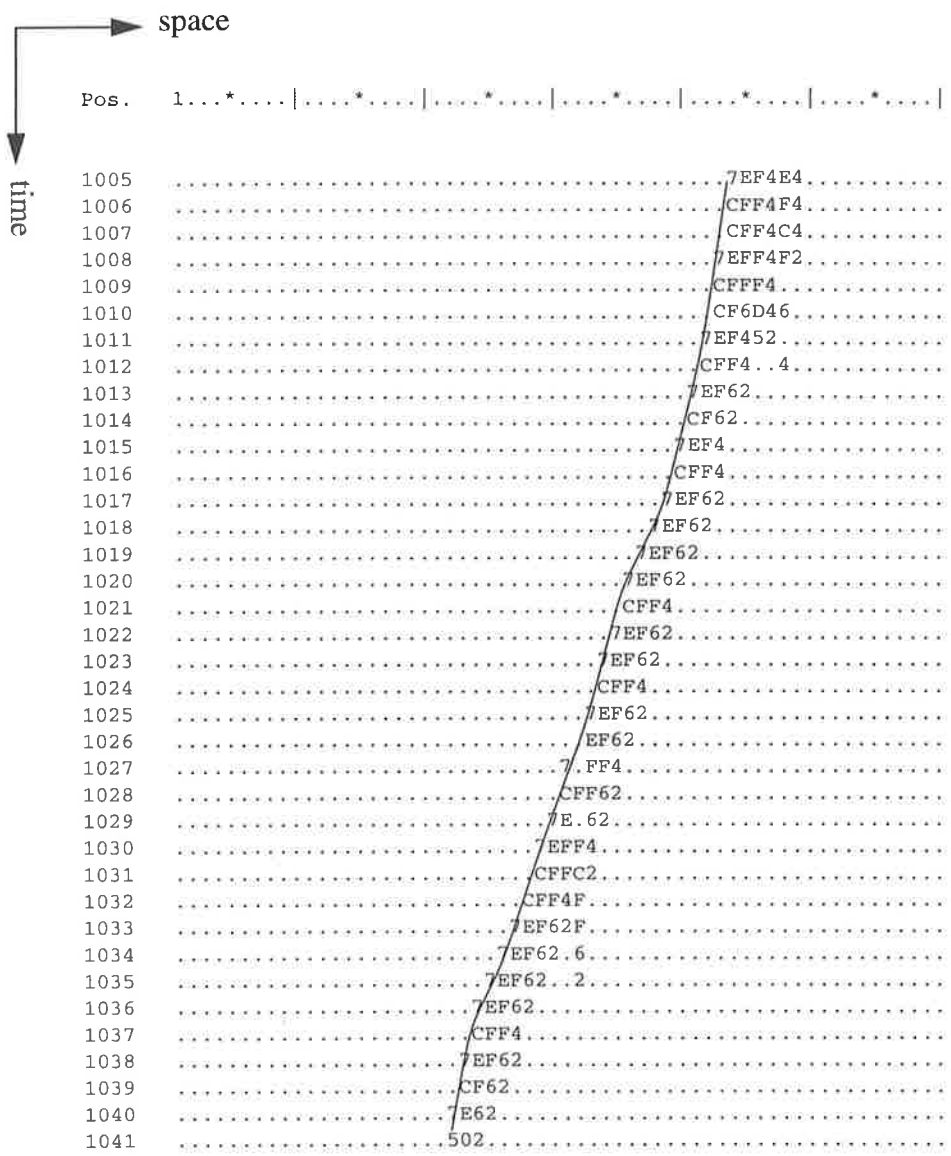


Fig. 7.18 Motion pattern of a person walking across the sensor's visual field at a distance of 3 to 4 metres.

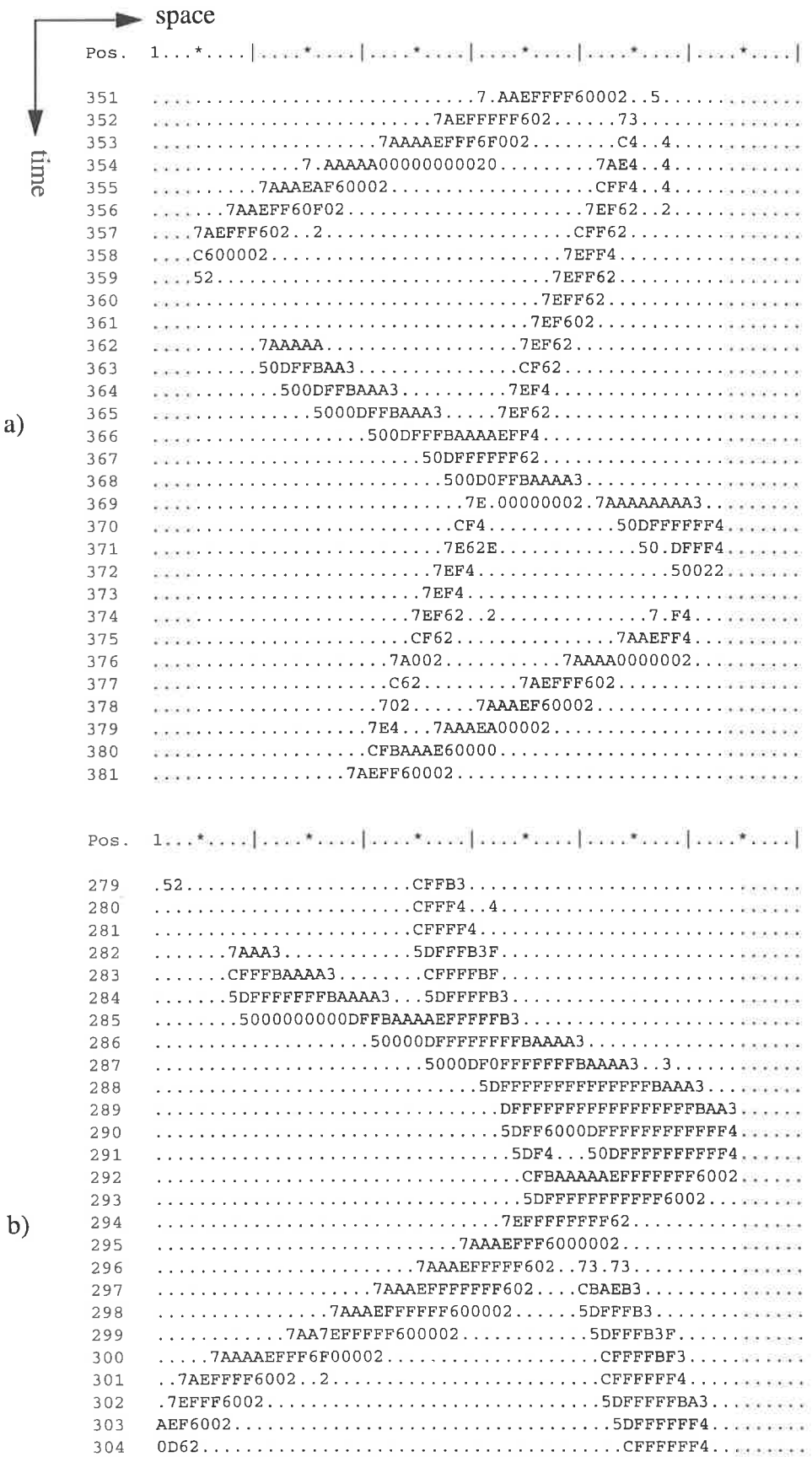


Fig. 7.19 Motion patterns of two objects moving simultaneously. a) Fast motion. b) Slow motion.

7.5 Summary

In spite of the detecting circuit's problems, due to the effect of changes in the transfer characteristics caused by circuit mismatches, the smart micro-sensor I has been successfully designed using mixed analog-digital CMOS VLSI technology. The sensor, which is simply equipped with a gradient index lens, works in real-time, albeit for moving edges which present an intensity decrease and preferably under natural light sources. This achievement nonetheless indicates that the principles of motion detection in insect vision can be reproduced into hardware using advanced VLSI technology. For a general comparison with other VLSI sensory devices, see, e.g., Yakovleff (1995).

Moreover, real-time experimental results have confirmed that simulated and real-time template responses are consistent. In particular, the following remarks obtained from the analyses in Chapters 4 & 5 are reinforced:

- Only a small number of templates occur due to motion, for example 15 templates frequently occur, compared with the total number of 81 templates, and only 4 templates are DMSTs indicating motion of dark-to-bright edges.
- The assumption that DMSTs and PCTs should occur in pairs is corroborated by the experimental evidence.
- Spatial and temporal templates appear on vertical and horizontal segments of motion trajectories respectively.
- In the spatio-temporal domain, motion trajectories show the shape of a staircase as illustrated in Fig. 7.17, and are characterised by the conjugate pairs of Table 7.2. However, position conjugate templates are not always reliable to represent staircase motion trajectories. Instead:
- Directionally motion-sensitive templates can be used robustly as features for representing motion trajectories.
- The sensor has a detecting range of from 25 centimetres up to 3 to 4 meters, which is reasonably useful for mobile robots, for instance, in detecting obstacles for navigation, and the sensor can work in dynamic environments where several objects may be moving simultaneously.

Chapter 8

Smart Micro-Sensor II

8.1 Introduction

The smart micro-sensor I, described in Chapter 7, was the first implementation of the research. The main mission of that implementation was to prove that the principles of insect vision incorporated in the template model, are physically realizable. Although sensor I detects motion of dark-to-bright edges in a real-time environment, there are a number of disadvantages in that design, for example, the directional information provided by the directionally motion-sensitive templates was not explicitly used (see Chapter 7). Furthermore, the performance of sensor I may be affected by artificial lighting conditions, and hence reduce its suitability for different environments. In short, the design is not optimal in terms of the utilisation of the directional information, and is not efficient in terms of its adaptation to different conditions.

One way to overcome these problems is to go back to the second level of the Marr methodology, where the algorithm and the data representation should be analysed and organised in an efficient manner. Indeed, the template model was formally re-evaluated in Chapter 4, where it was shown that it is functionally similar to the well-known Reichardt model, and moreover, has a few advantages over the Reichardt model. Other additional benefits which are obtained from the feature representation of motion trajectories using the DMSTs were also exploited (Chapter 5). These advantages and benefits are now reflected in the smart micro-sensor II (sensor II for short), whose primary objective is to provide a test-bed for some analog circuits and for the implementation of different velocity estimation algorithms (Chapter 6). In particular, the backward tracking algorithm is chosen for the implementation in this design.

The implementation of the smart micro-sensor II is the focus of this chapter. Some of the material contained in this chapter was reported in Nguyen et al. (1996). Section 8.2 firstly describes the overall architecture of sensor II while Section 8.3 shows the timing of sensor II. The next three sections describe three constituting processors: front-end, template, and velocity estimation. Section 8.4 describes briefly the front-end processor, which was

previously reported in Moini et al. (1995a, 1995b). Section 8.5 and Section 8.6 present in detail the template and velocity estimation processors respectively. Section 8.7 provides a comparison between the two sensors I & II, which highlights major differences in their designs. Section 8.8 discusses the testing of the constituting processors and provides experimental results of velocity estimation using sensor II to detect moving objects in real time. Finally, the chapter is summarised.

8.2 General Description

In general, the functionality of the smart micro-sensor II is identical to that of the smart micro-sensor I, where the three separate stages corresponding to the three essential tasks, which are detection, template interpretation, and velocity estimation, are preserved. However, sensor II is not as monolithic as sensor I in the sense that sensor II is implemented in different components using two different semiconductor technologies. The front-end processor, which comprises analog circuits for detecting changes in light intensity, is a custom VLSI design using 1.2 μm CMOS technology. Using Erasable Programmable Logic Design (EPLD) technology from Altera, the template and velocity estimation processors are

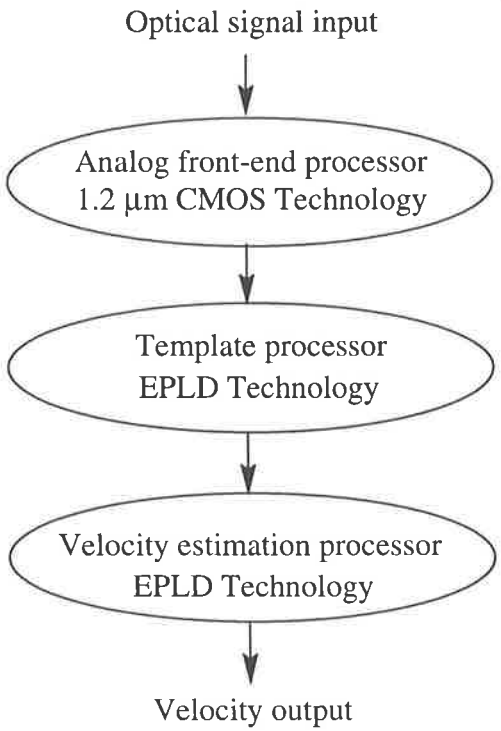


Fig. 8.1 Basic functional blocks and technologies for implementing sensor II.

implemented digitally into three separate components. Separating the analog and digital circuits presents a number of practical advantages. This particular architecture allows the design of the front-end processor to emphasize the characterisation of analog circuits, and to enhance the performance of the early processing of visual motion. Furthermore, the design of the digital processors using EPLDs allows the design to be reconfigurable and is economical, since the design cycle using EPLDs is short. In particular, EPLD technology is suitable for implementing different high-level image processing algorithms such as those presented in Chapter 6. Fig. 8.1 depicts the functional blocks and the technologies for implementing sensor II.

8.3 Timing

The timing diagram is shown in Fig. 8.2. In principle, the sequence of operations of the processors is identical to that of sensor I within a sampling interval. The frond-end analog processor is sampled first, then the template processor combines intensity change data into templates, and lastly the velocity estimation processor computes velocities of detected moving objects.

Although a sampling frequency of 100 Hz should be suitable for real-time applications

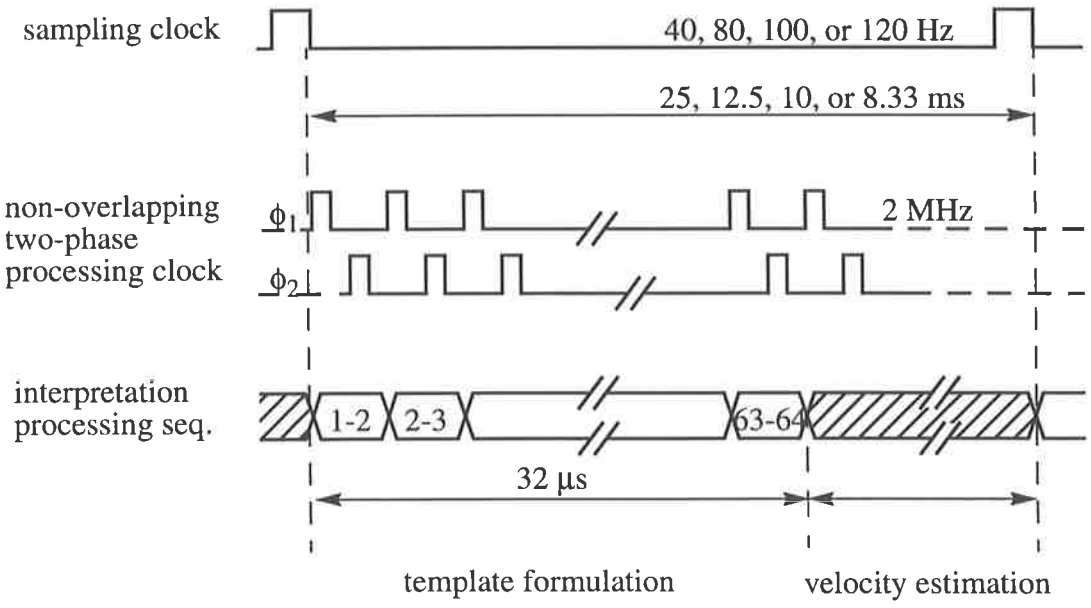


Fig. 8.2 Timing diagram for the smart micro-sensor II with 2 Mhz processing frequency.

(Chapter 7), sampling may also be accomplished at 40, 80 100, or 120 Hz for testing purposes. Finally, an non-overlapping two-phase clock is used in both the template and velocity estimation processors (ϕ_1 is phase one and ϕ_2 is phase two). The clock signal is provided externally, and may therefore be set at a higher frequency than the 2 MHz as shown Fig. 8.2.

8.4 Front-End Processor

The front-end processor comprises 64 parallel detecting channels † with special analog circuits. Each channel has four major analog components: photodetector, signal enhancement, temporal differentiator, and thresholding circuit. The architecture of this processor is illustrated in Fig. 8.3.

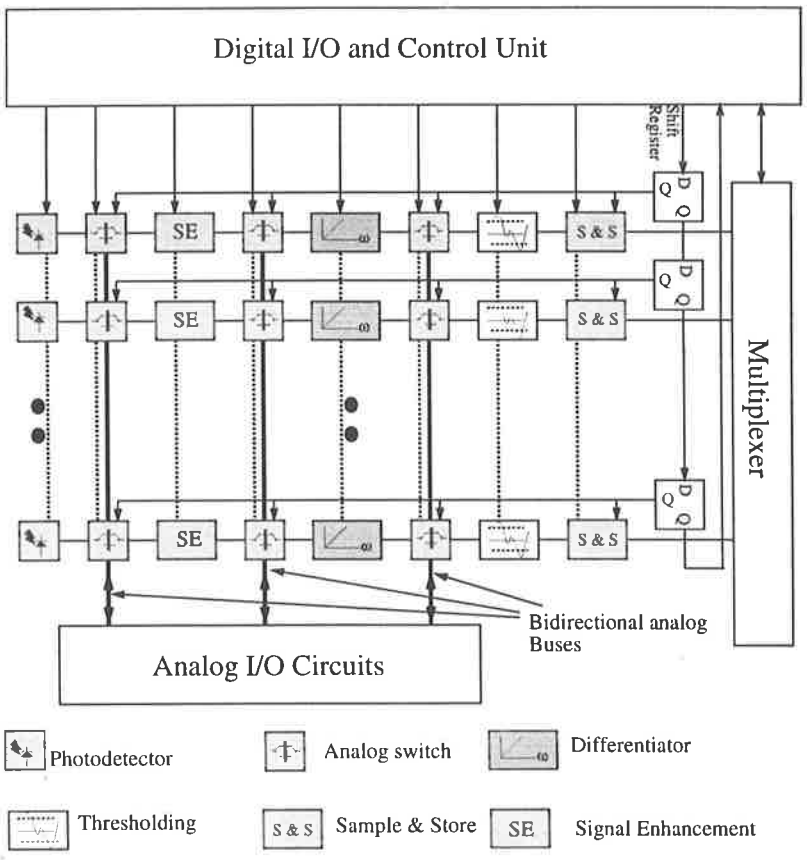


Fig. 8.3 Schematic of the front-end processor.

†. Sixty four photodetectors are used in sensor II in order to facilitate the digital addressing scheme. However, the optical resolution is the same as that of sensor I, i.e., the size of and the separation between the photodetectors are unchanged.

The photodetectors as shown in Fig. 8.4 are designed to operate in two different modes. One mode consists of using the *pn* parasitic junction diode as in sensor I. The other uses the bipolar junction transistor (BJT), which has a higher sensitivity level than that of the diode. The main benefit of two-mode scheme is to increase the dynamic range of the sensor. In fact, such an architecture is similar to that of the human retina, where the bipolar transistors correspond to the rod cells, which are responsible for *scotopic* vision, i.e., the visual response at low levels of illumination. The junction diodes correspond to the cone cells, which are less sensitive than the rods, and are responsible for *photopic* vision, i.e., the visual response at levels of illuminance 5 to 6 orders of magnitude higher (well-lighted room or bright sunlight - Levine 1985; Jain 1989). The conversion between the two modes is affected through a control signal V_c which turns on or off a MOS transistor. When V_c is low, the base and the emitter of the BJT are disconnected and the BJT is effectively operating. When V_c is high, the base and the emitter are shorted and the p-well effectively becomes the p-part of the pn junction diode, while the BJT collector becomes the n-part of the diode.

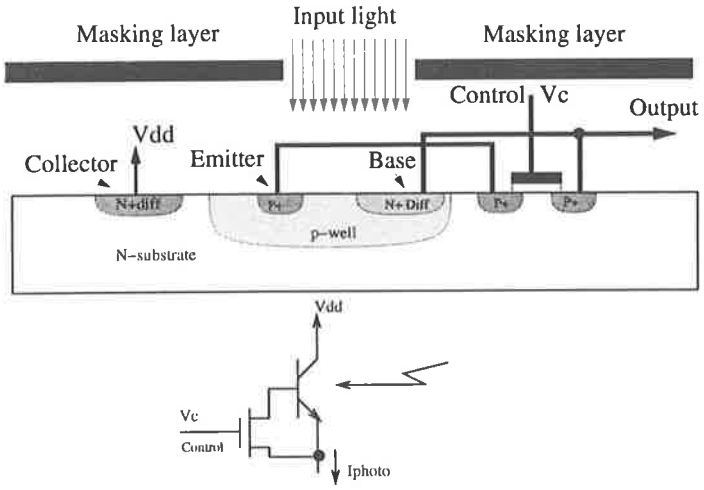


Fig. 8.4 The physical structure and schematic representation of the photodetector.

The role of the signal enhancement component (Fig. 8.5) is to filter out noise picked up from the environment, such as the 50Hz signal generated by fluorescent lights. Noise reduction is accomplished by spatial filtering and division. The signals from the photodetectors in a local neighbourhood are averaged using a current-mode spatial averaging circuit, which is then followed by a circuit dividing each channel's signal by the resultant average signal, in order to remove any multiplicative noise. Therefore, this component is referred to as *multiplicative noise cancellation.*

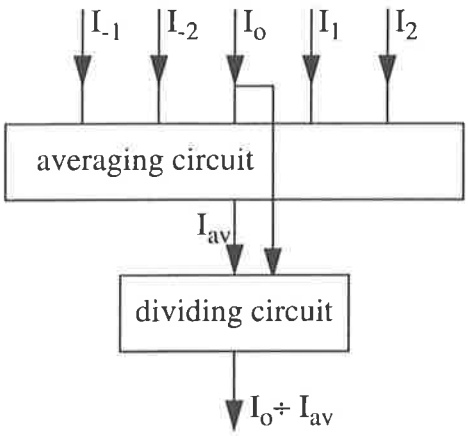


Fig. 8.5 Schematic representation of the signal enhancement component.

The third component of the front-end processor is the differentiator used for detecting changes in intensity. Similar to the differentiator used in sensor I (see Fig. 7.6), this differentiator also has a feedback configuration with an active resistor, although the later may be selected in two different modes incorporated in the active resistor. The active resistor may thus be set to two significantly different values of the resistance, which effectively enhances the dynamic range of the detecting channels.

Lastly, the thresholding circuit converts the intensity changes into two digital signals. This circuit provides adjustable thresholding levels, which can be set externally. The adjustability of the thresholding levels makes the sensor potentially suitable to a wide range of environments since the thresholding levels can be adjusted to be compatible with the luminance of an environment. At each sampling instant, the two-bit codes of the intensity change states from the 64 detecting channels are serially shifted out from the processor by using a parallel-to-serial shift register.

It is also worth noting that switching circuits and analog buses are employed between any two components for testing purposes. These switches provide bidirectional analog signal transfers over an analog bus, and can be selected so that the output of each component may be probed. For more details, see Moini et al. (1995a, 1995b).

8.5 Template Processor

The template processor is an engine which forms templates as eight bit codes, and then filters out motion-insensitive templates. The analyses of Chapters 4 & 5 reveal that only eight directionally motion sensitive templates (DMSTs) are sufficient for detecting moving objects and for velocity estimation. Hence, only four bits are needed for encoding the eight

DMSTs and a single encoding which represents all motion-insensitive templates. In order to facilitate the hardware design of the velocity estimation processor, it may be desirable to encode the DMSTs in such a way that switching from one property to another only requires changing a single bit. In this design, the first bit is used for indicating motion, the second for distinguishing between a pair of DMSTs (Section 4.6.2), the third for representing the direction of motion, and the last for the polarity of object edges as shown in Table 8.1. Other templates are encoded as 0000 binary.

Values	A_3	A_2	A_1	A_0
1	motion	outer	left	dark-to-bright
0	no motion	inner	right	bright-to-dark

Table 8.1 The format of 4-bit template codes.

As the output of the analog processor is a two-bit wide serial sequence of intensity changes at each sampling instant, the template processor should have an appropriate storage so that the serial sequence of intensity changes at a current sampling instant is stored for forming templates at the next sampling instant. During the sequential transfer of the outputs of the detecting channels, the two-bit output of a current channel should also be latched for use with the output of the adjacent channel to form a template. The most suitable architecture for this operation is shown in Fig. 8.6. A serial shift register is employed to store the sequence of intensity changes. The shift register has 64 cells, each of which has 2 bits,

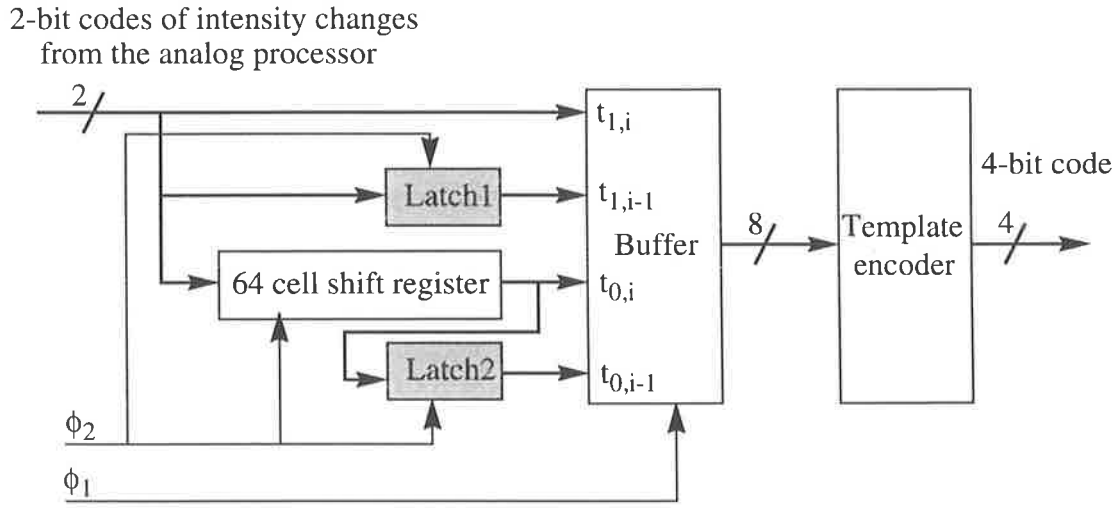


Fig. 8.6 The architecture of the template processor.

to accommodate the intensity changes of the 64 detecting channels. Two latches, 1 and 2, are used to temporarily store the data of the adjacent channel at the current and previous time instants respectively. The non-overlapping two-phase clock is very useful in this operation. One of the phases, for example ϕ_1, is used in the template forming operation, while ϕ_2 is used to store data into storage elements. A formed template is then fed to an encoder which encodes each eight-bit template into a 4-bit code.

Four-bit template codes are then stored into a RAM which accumulates template data at 8 consecutive sampling instants as discussed in Section 6.3.2. Each row has 64 four-bit cells corresponding to 64 detecting channels. A pointer is used for the row address, and is incremented in a wraparound counter so that the memory is accessed as a FIFO register as shown in Fig. 8.7.

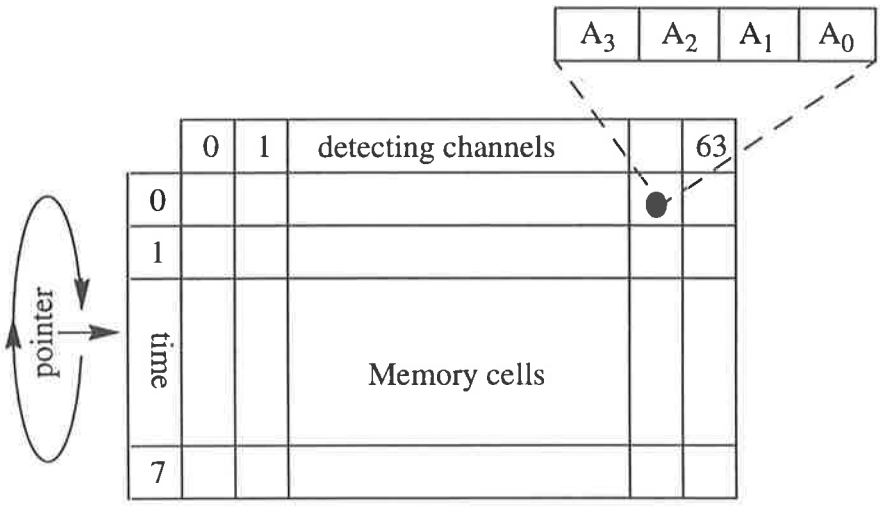


Fig. 8.7 The memory architecture.

8.6 Velocity Estimation Processor

The velocity estimation processor performs the backward tracking algorithm in a sequential fashion of search-and-track for motion trajectories. The algorithm is divided into two main stages: *searching* and *tracking*. These two stages are repeatedly carried out across the row of the 64 detecting channels at each sampling instant. In the searching stage, the processor scans across the currently stored row of 4-bit template codes in the RAM to find out if any DMSTs are present. The processor reads a code at each memory cell and checks whether the template is motion-sensitive or not. If a DMST is found, the processor subsequently performs the tracking stage, where the processor looks for any possible motion

trajectory which may start from that cell. The processor searches for occurrences of the same DMST in the previous sampling instants in a local neighbourhood of the currently detected DMST. At the end of the tracking stage, the processor reports velocity information, i.e., the travelled distance and the corresponding taken time. Thus, the processor provides logical operations to handle the detection of DMSTs, and arithmetic mechanisms to compute space and time projections of a motion trajectory.

In terms of hardware architecture, the argument above suggests that the processor should have at least four main modules referred to as *matching*, *distance-travelled, time-taken*, and *controller* as shown in Fig. 8.8. The matching module handles the detection of DMSTs. The distance-travelled module measures the distance travelled in a motion trajectory. Similarly, the time-taken module calculates the corresponding taken time from the trajectory. The operation and communication of the processor are governed by a processor controller. In order to avoid the racing problem † between control signals and logical signals, ϕ_1 is used in the controller, while ϕ_2 is used in other functional modules.

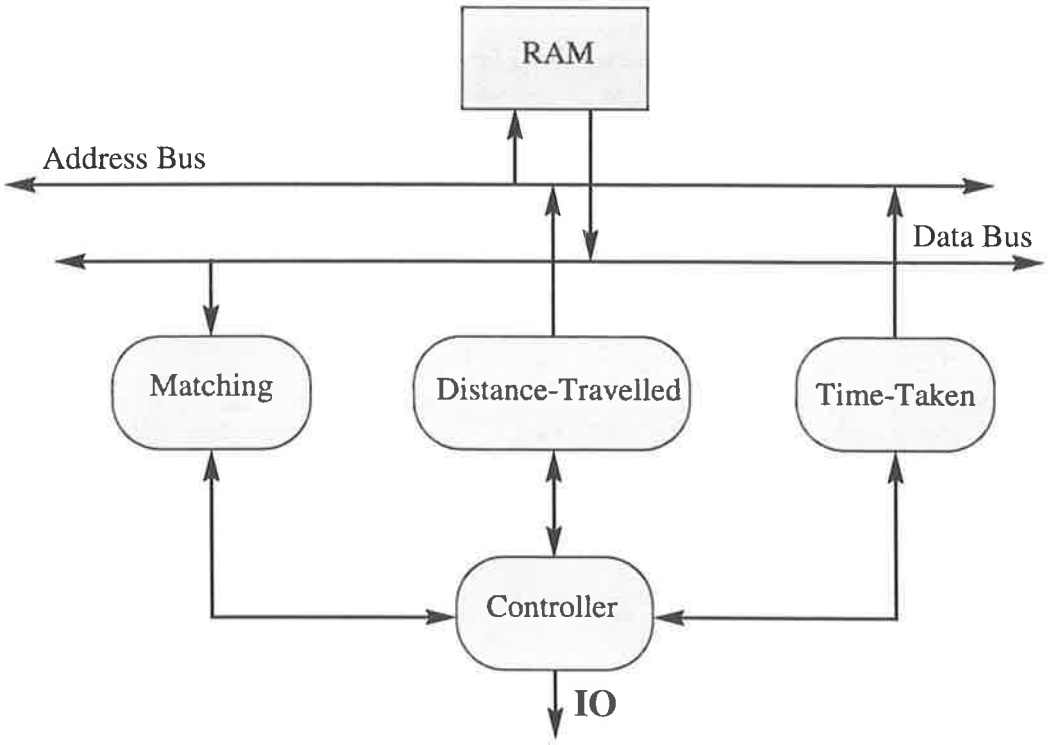


Fig. 8.8 The architecture of the velocity estimation processor with an external RAM.

†. In digital devices, control signals should activate the devices after logical signals' arrival. If the incoming order of the two kinds of signals is reversed, a racing problem may occur, and the devices may produce erroneous outputs.

8.6.1 Processor Controller

The processor controller handles all communication between any of the functional modules within the velocity estimation processor. The controller carries out its tasks by generating control signals to either activate or de-activate the functional modules.

Although there are a number of techniques for designing a controller, the most common technique is to break an algorithm into a finite sequence of operating states and form a so-called *finite state machine* for a designed problem. The finite state machine provides an organised structure for capturing control sequence and operation. For this particular velocity estimation processor, there are a total of 10 states, which are defined in Fig. 8.9.

Initially, the controller stays in state 0 until the processor-enable signal becomes active (1). It then goes to state 1 to initialise registers and to store the current row address. Subsequently, the processor is in the searching stage where states 2 and 9 are alternately active to enable the processor to read a template from the memory, and then to check whether the template is motion-sensitive. These two states allow the processor scanning across the currently stored row of templates to find if any DMSTs are present. A detected DMST and its current address are stored into template and location registers in state 3 respectively. The stored DMST then becomes the target template which will be searched for during the tracking stage. The stored address becomes a reference point for further searches of the target template.

The processor then enters the tracking stage controlled by four states: 4, 5, 6, and 7. State 4 handles a new search window in a previously stored row of template, that is, a search window is established one row above the current row. State 5 is to address memory cells in that new search window. The memory cells of the search window are addressed in state 5. For each address issued, the processor reads a template and compares it with the target template (state 6). If there is a match, the processor stores the current location of the target into the second location register. These four states are repeated until the end of a motion trajectory or the memory boundary is reached, and the processor reports velocity information in state 8.

8.6.2 Matching Module

The main function of the matching module is to check whether or not a template is motion-sensitive in the searching stage and is a target template in the tracking stage.

In the searching stage, as the most significant bit (A_3) of the DMSTs is high, detection consists of AND-ing A3 with a search-enable signal (state 2). If the template is directionally

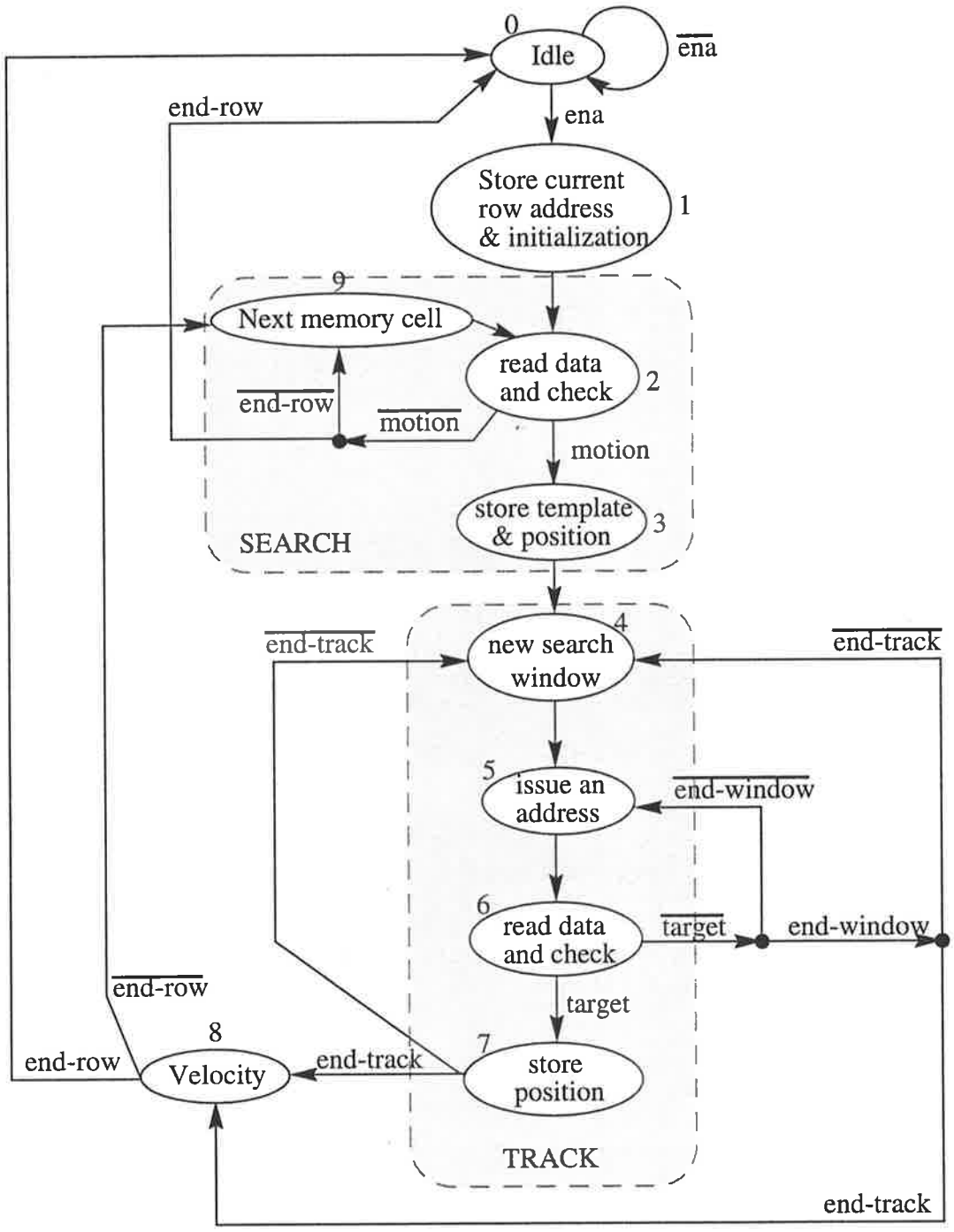


Fig. 8.9 The finite state machine of the velocity estimation processor.

motion-sensitive, the AND gate output is high, and is used as a *motion* signal to store the template into the *target-template* register, and its direction of motion, A_1, into a *L-R-flag (left-right)* register.

In the tracking stage, a template is compared with the known target via a *comparator* which comprises exclusive-OR gates and an AND gate. The comparator is activated by a track-enable signal (state 6). Fig. 8.10 depicts the architecture of the matching module.

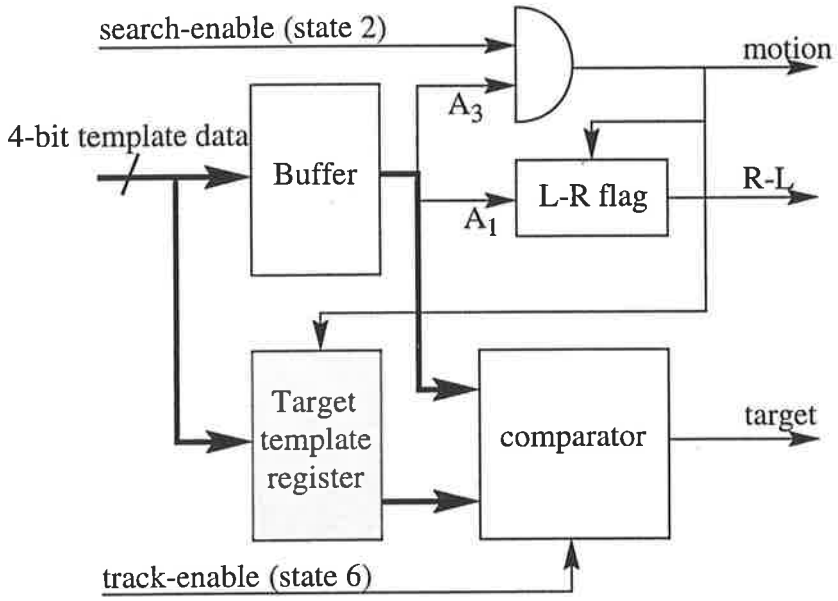


Fig. 8.10 The architecture of the matching module.

8.6.3 Distance-Travelled Module

The distance-travelled module manages searching activities on the spatial axis of a motion trajectory. These activities include handling search windows and column addresses, and measuring the distances travelled by an object between sampling instants. The architecture of this module is shown in Fig. 8.11, where gray blocks are registers, black blocks are counters, and others are some logic functions.

In the searching stage, this module starts searching for DMSTs with the aid of a *channel address generator*, which is a 6-bit counter, and which generates channel addresses across the memory. When a DMST is found, its spatial location is written into both *target-position* and *initial-position* registers. The initial-position register keeps the current position of a DMST, i.e., a moving object, while the target-position register is used to store the current position of the detected target, i.e., the known DMST. The address generator gets access to the column-address bus during this stage.

In the tracking stage, an *up/down* counter, whose initial value is set to the value of the target-position register, issues channel addresses. The up/down function of this counter is set by the direction of motion, which is specified by the value of the L-R flag. If the L-R flag is high, that is, an object moves to the left, this counter is set to a down function to search to the right, and vice versa. A multiplexer is employed to switch the up/down counter to the column-address bus during this stage. In case of a target hit, the location of the target overwrites into the target-position register as the current value, and a new searching window is set up until the end of the trajectory is reached. If the target is not found within the current window, the processor will search for the target in the same spatial neighbourhood but at earlier time instant.

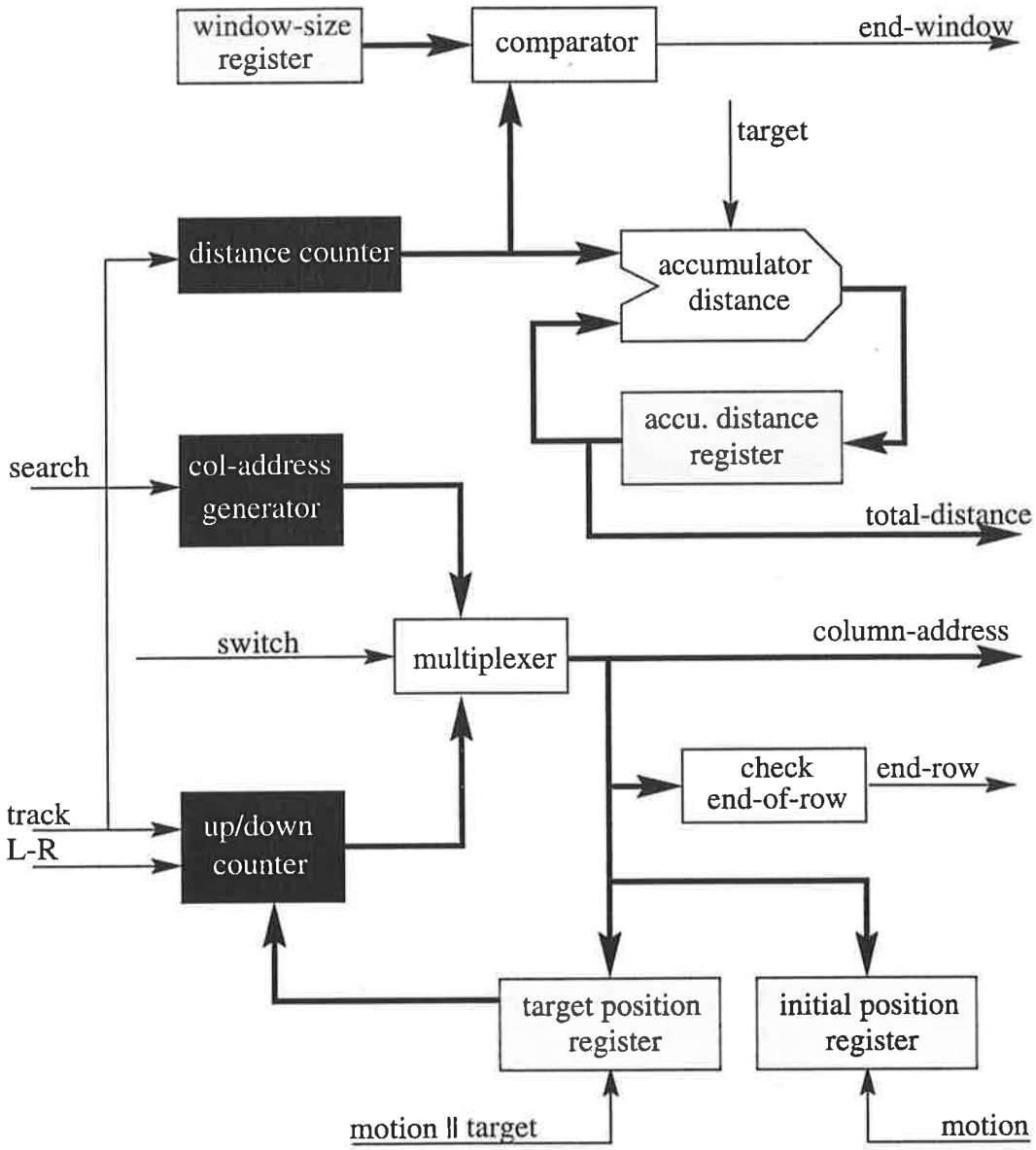


Fig. 8.11 The architecture of the distance-travelled module.

The size of the search window is pre-defined and stored in a *window-size* register. The value of this register specifies the maximum distance which a target can move in one sampling interval. The search area is monitored by a *distance* counter which is increased by one each time a template is read from the memory to the processor. The distance counter's value is compared with the value of the window-size register in order to signal the end of the search window. This counter is also used to measure how far the target moves between two consecutive sampling instants. In case of a target hit, the value of this counter is accumulated into a *distance-accumulating* register. At the end of the tracking stage, the value of the distance-accumulating register is the total distance travelled by the target during the total time taken measured by the time-taken module.

8.6.4 Time-Taken Module

The time-taken module is a companion to the distance-travelled module, and manages activities on the temporal axis of a motion trajectory. These activities include handling row addresses, and measuring the time taken for a target to move from one location to another. The architecture of the time-taken module is illustrated in Fig. 8.12, where there are two registers illustrated as gray boxes, two counters as black boxes, one multiplexer, and one logic unit.

When the velocity estimation processor is activated, the current row address is stored into a *row-address* register. A *down* counter, which is initialised to the current row address,

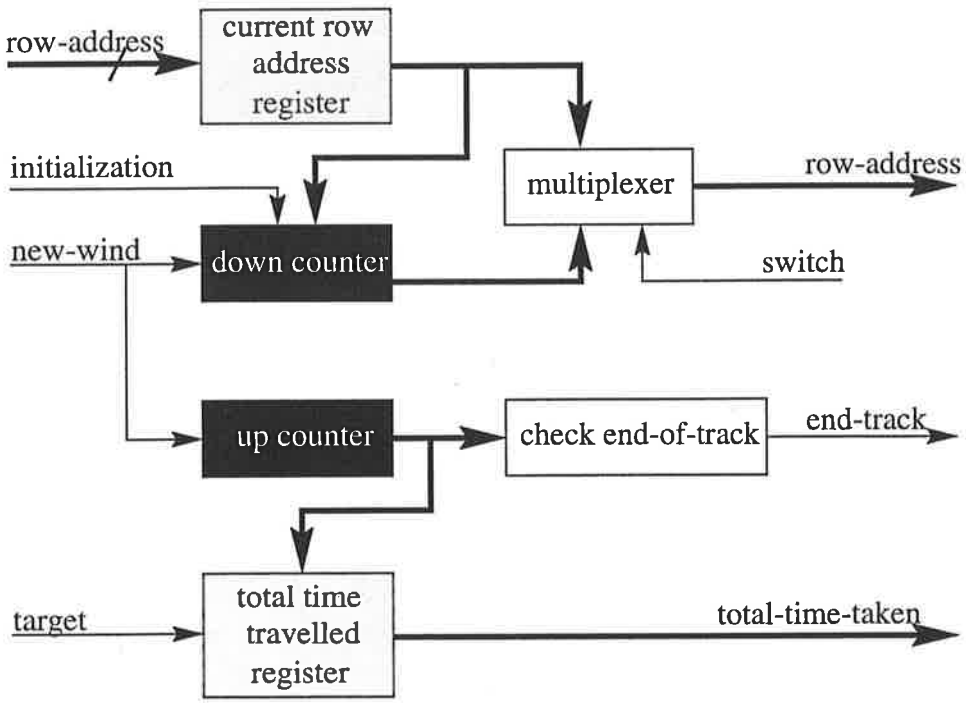


Fig. 8.12 The architecture of the time-taken module.

is employed to manage row addresses during the tracking stage. When a *new-wind* signal becomes active the down counter is decreased by one indicating that a new search window has been established. Similar to the distance-travelled module, a multiplexer is employed to switch either the current address register, or the down counter, to access the row-address bus.

To avoid the need for a subtractor to calculate the temporal projection of a trajectory from its two end points, an *up* counter is used instead. This up counter is initialised to zero each time a track is begun, and is incremented each time a new search window is set up. Once a target is hit, the value of the up counter is then written into a *total-time* register as the time taken for a target to move from the initial position to the current position. At the end of the tracking phase, the value of the total-time register is the total time taken for a target to move along a part of its trajectory, which has just been tracked.

8.7 Comparison between the Designs of the two Sensors

Objectively, the design of sensor I was meant as a proof of concepts, that is, to prove that the principles of motion detection of biological vision systems can be applied to the design of machine vision systems using advanced electronic concepts. Sensor I is a monolithic and fully custom VLSI design using 2 μm CMOS technology. On the other hand, sensor II is aimed at providing a test-bed for all associated analog circuits and for the implementation of different velocity estimation algorithms. Thus, the latter is more flexible in terms of testability because it is constructed from separate modules and implemented in different technologies.

In the context of timing, the main difference between the sensors is the processing frequencies in the template and velocity estimation processors. In sensor I, two different frequencies of 1 MHz and 4 MHz are used in the template and velocity estimation processors respectively (Section 7.3.2), but in sensor II, only one frequency of 2 MHz is employed in both the template and velocity estimation processors.

At the front-end stage, sensor I is designed with a fixed pn junction photodetector, a single-mode active resistor, and a level-fixed thresholding circuit. In contrast, sensor II provides flexibility and advanced design with a two-mode photodetector (i.e., a pn junction diode and a bipolar transistor), a mechanism for multiplicative noise cancellation, a spatial averaging circuit, a two-mode active resistor, and a level-adjustable thresholding circuit.

At the template interpretation stage, sensor I has a bulky template RAM compared to a compact and smart encoder in sensor II. The template RAM must be initialised each time

the sensor operates, but the encoder is fixed and encodes DMSTs only.

At the velocity estimation stage, the design of sensor I followed an ad hoc approach which resulted in an inefficient processor comprising six identical tracking engines (Section 7.3.5). Each of the engines is responsible for detecting up to moving objects in its tracking region. However, these engines do not handle the case where objects may be moving across boundaries between two regions. Furthermore, the velocity estimation processor does not utilise the directional information contained in the templates to identify tracking direction. These drawbacks are avoided in sensor II, where the backward tracking algorithm is chosen for the implementation. Only one tracking engine is required and it detects moving objects within the whole visual field. Associated with the velocity estimation processor, a memory is needed for accommodating template data for eight consecutive sampling instants. Table 8.2 summarises the major differences between the two sensors I & II.

Functionality	Sensor I	Sensor II
Objective	proof of concepts, fixed design, fully custom design, 2 μm CMOS, compact.	test-bed, flexible design, not monolithic, different technologies: 1.2 μm CMOS and Altera EPLD.
Timing	two different frequencies: 1 MHz for forming templates, 4 MHz for processing template data.	one frequency: 2 MHz for both forming and processing templates.
Front-end	pn junction-based photodetectors, no multiplicative noise cancellation, no spatially averaging, one-mode active resistor, fixed thresholding levels.	pn junction- and bipolar transistor-based photodetectors, multiplicative noise cancellation, spatial averaging, two-mode active resistor, adjustable thresholding levels.
Template	reconfigurable RAM.	special encoder.
Velocity	ad-hoc approach, no use of information on directions of motion, several identical tracking engines, problem on crossing boundaries between tracking regions, no template memory required.	backward tracking algorithm, full use of information on directions of motion, one tracking engine, whole visual field, requires a template memory.

Table 8.2 A summary of the main differences between the two sensors.

8.8 Testing

8.8.1 Front-End Processor

Similar to the procedure for testing the smart micro-sensor I, the analog front-end chip is interfaced with a Pentium-based personal computer, which is equipped with a National Instruments Data Acquisition (NI-DAQ) interface to handle communication between the chip and the computer. A gradient index lens is glued on the chip's surface using a special ultra-violet-hardened optical glue.

The chip is tested for each of the analog components. The sensitivity of the transistor-based photodetector is 300 times higher than that of the diode-based one over an approximate input range of 80dB as shown in Fig. 8.13. The signal enhancement component has the effect of enhancing edges and of filtering noise. For more details of testing and results, see, e.g., Moini et al. (1995b).

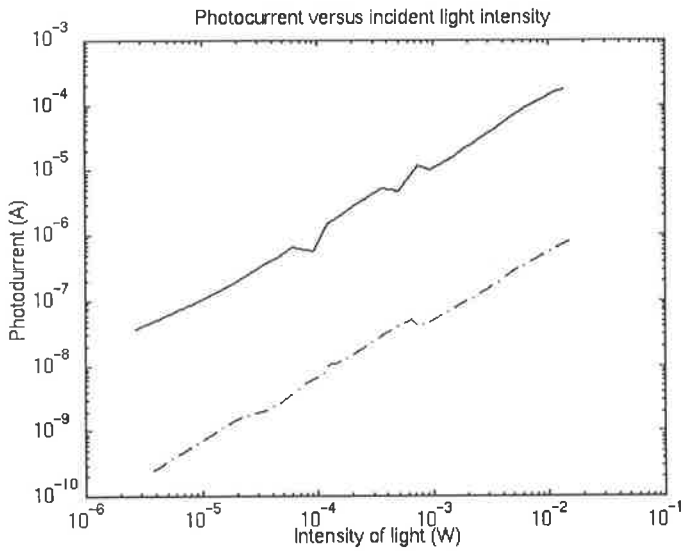


Fig. 8.13 Test results of the two-mode photodetector.

Admittedly, there still are some problems with the analog differentiators in this design, which have not been yet resolved. The outputs of the differentiators are very sensitive to biasing currents. The differentiators also behave in a very strange manner in the sense that they all respond simultaneously to a local intensity variation occurring across only a few channels. Moini, who designed this chip, suspects that these problems might be caused by a serious problem of charge leakage between the isolated differentiators. As a result, the integration of the constituting processors on a single chip to form a complete system is inhibited.

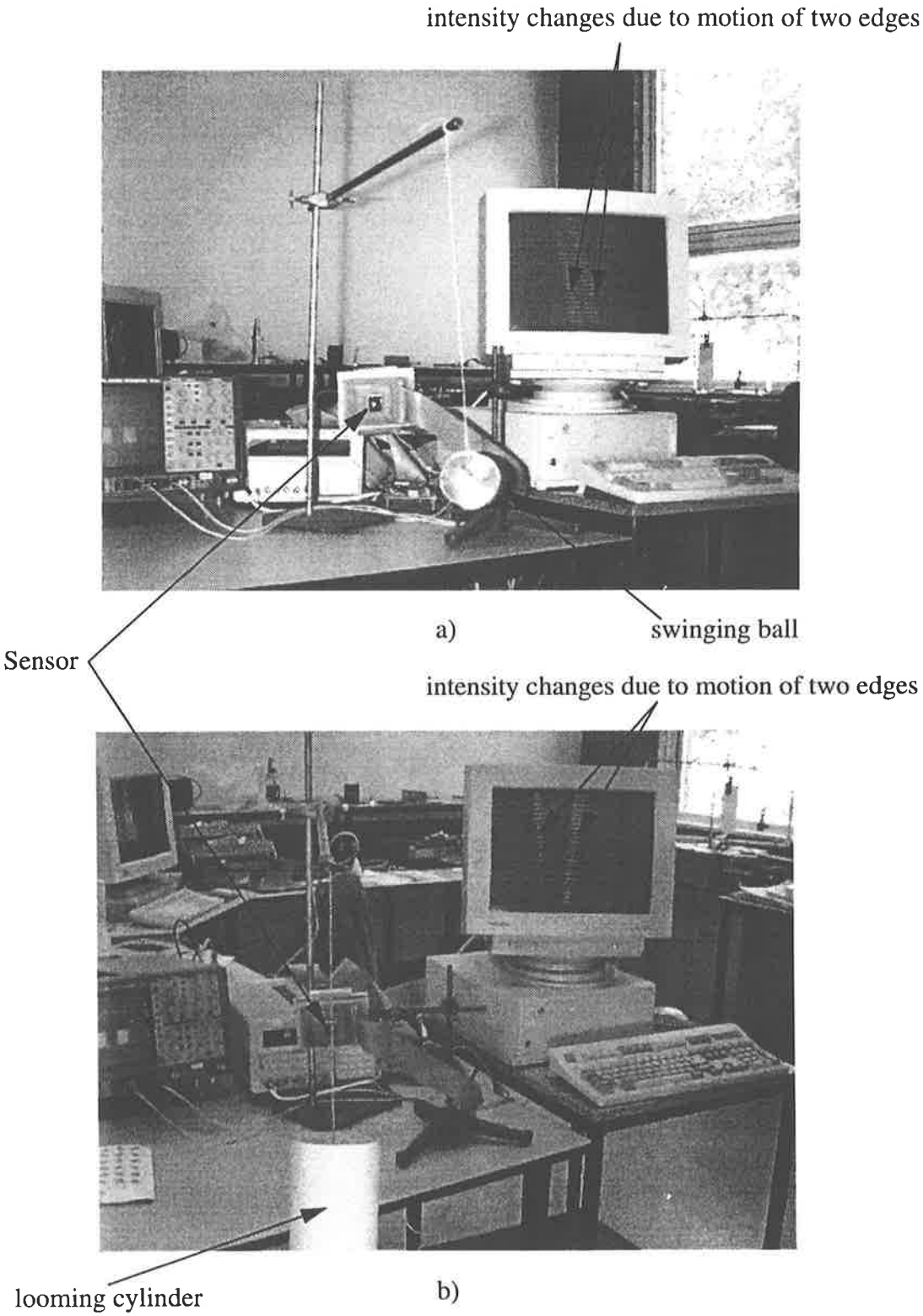


Fig. 8.14 Real-time experiments. a) Intensity changes due to motion of the two edges of a swinging ball which is moving to the right. b) Intensity changes due to receding effect of cylinder which is moving away from the sensor.

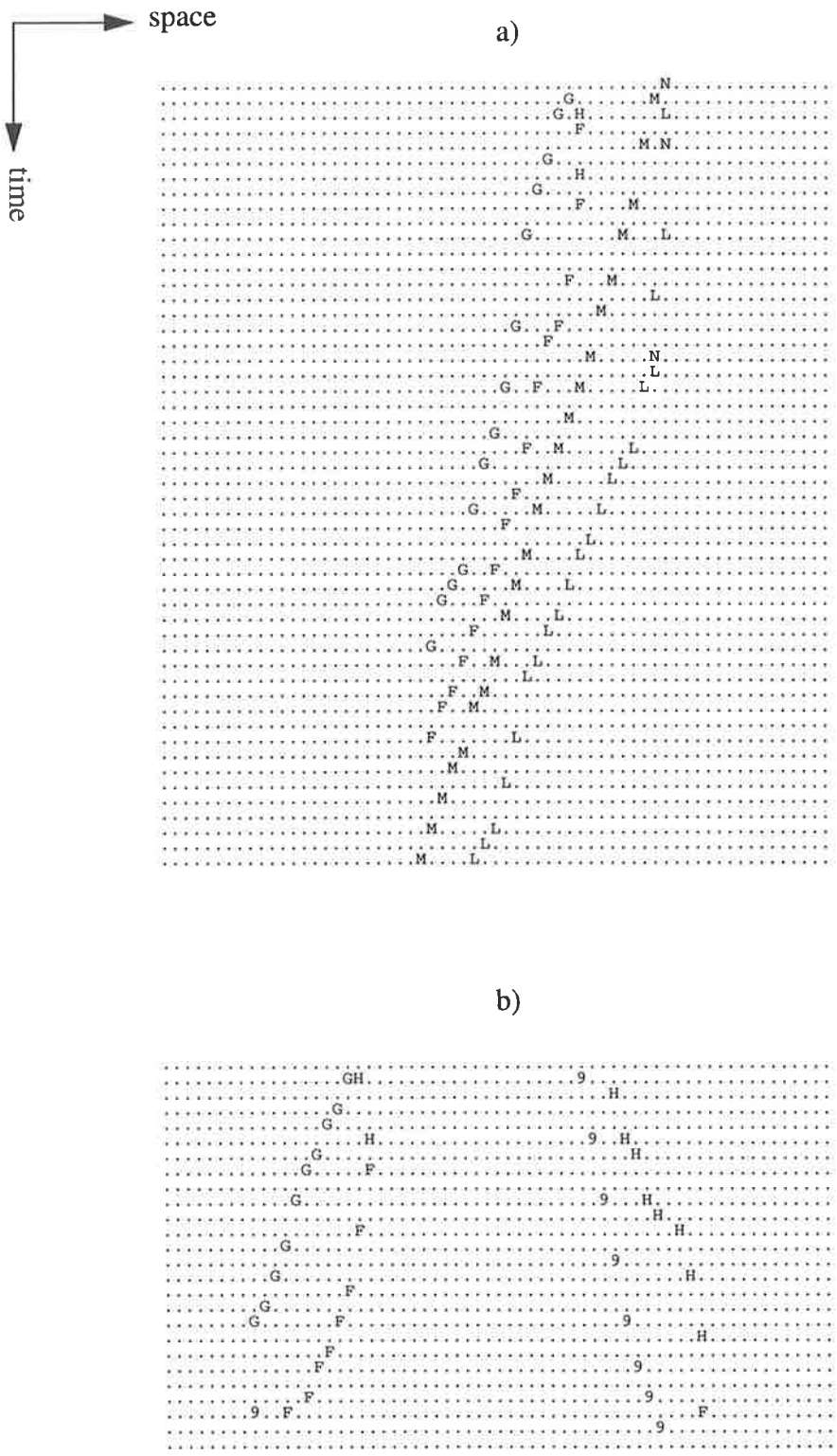


Fig. 8.15 Motion trajectories represented using DMSTs. a) Swinging ball moving to the left. b) Cylinder moving away from the sensor.

One effective way to solve the problems of the analog differentiators is to intensively study their characteristics by designing and testing different differentiation circuits. In fact, such a chip containing a number of test differentiators has been designed and fabricated; testing is also under way (Moini, private communication).

Despite the problems of the analog differentiators, it is still possible to read out the photocurrent due to the testability of the design. This current is amplified using a single stage bipolar transistor, and then further processed by a C program, which performs other functions of the front-end chip, i.e, multiplicative noise cancellation, differentiation, and thresholding. This testing procedure allows us to test the photodetector of the front-end chip with various stimuli in real time.

The experiments were carried out inside a laboratory with a white wall as the background. In order to be transferred to the outside, the photocurrent should be in a range of tens to hundreds of microampres, to cope with leakage and charging of some parasitic capacitors, such as those of transmission lines and of the chip's pads. To generate the photocurrent in that range, strong lighting conditions such as daylight at noon or light from nearby neon lamps were required.

For experimental purposes, a ball swinging from left to right, and a cylinder moving forward and backward are used (Fig. 8.14a and Fig. 8.14b respectively). These objects' motion trajectories, which are represented using DMSTs, are shown in Fig. 8.15. The results confirm that motion of each edge of the object can be recognised by a pair of DMSTs, one occurring on the virtual leading edge of the differential pulses, and the other occurring on the virtual trailing edge of the same pulses (see Section 4.6.2).

8.8.2 Template and Velocity Estimation Processors

The test is to verify the functional performance of the template and velocity estimation processors when they are combined. The input data to the template processor are a spatio-temporal image of changes in intensity, which is obtained from the test of the front-end processor, e.g., an intensity change pattern elicited by a moving hand as shown in Fig. 8.16a. The data are converted into a suitable format required by the Altera simulator. The outputs of the template and velocity estimation processors show detected DMSTs, distance travelled and corresponding time taken, e.g., the columns marked as TEMP, DIS, and TIME in Fig. 8.16b and Fig. 8.15c.

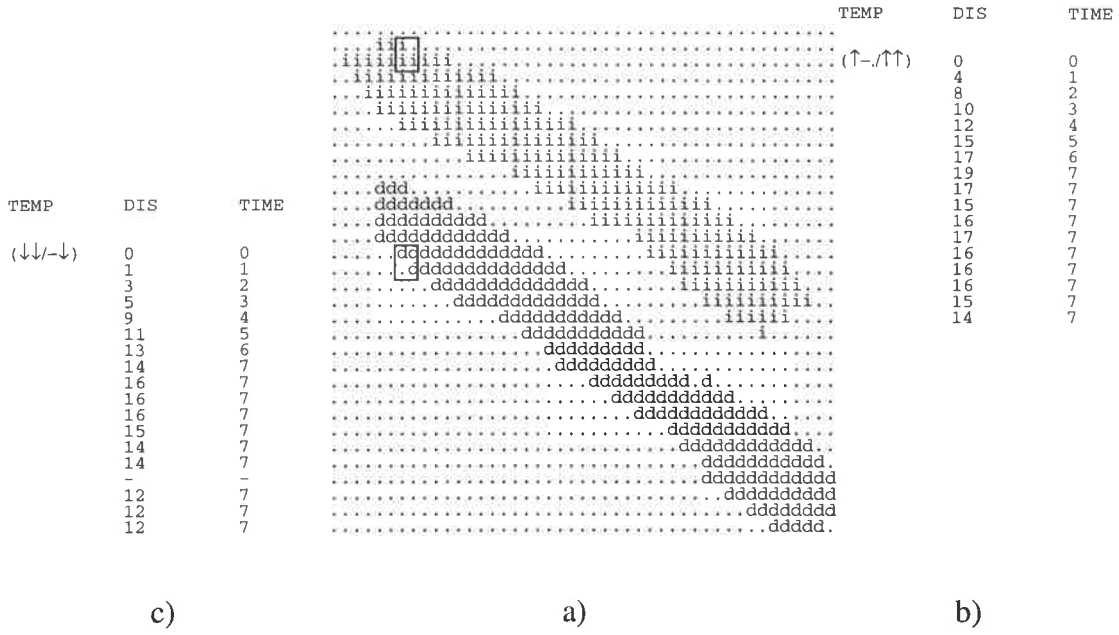


Fig. 8.16 Experimental results of the test of the template and velocity processors. a) Intensity-change input to the processors, where 'i' is an increase, and 'd' is a decrease in intensity. b) The DMST (↑–/↑↑) is detected and processed to obtain velocity information as travelled distances and corresponding times. c) Similarly, velocity information obtained from the search for the DMST (↓↓/–↓).

8.8.3 Velocity Estimation

This sub-section is devoted to real-time experiments of velocity estimation using the front-end chip as the sensor to detect moving objects whose velocities can be physically measured.

8.8.3.1 Apparatus

The apparatus used in these experiments is shown in Fig. 8.17. It consists of a rotating drum whose rotation is controlled by a step motor. The angular velocity of the drum can be measured fairly accurately with a frequency meter. The sensor is located approximately at the centre of the drum. An object is fixed to one of the four arms of the drum at a distance of 30cm.

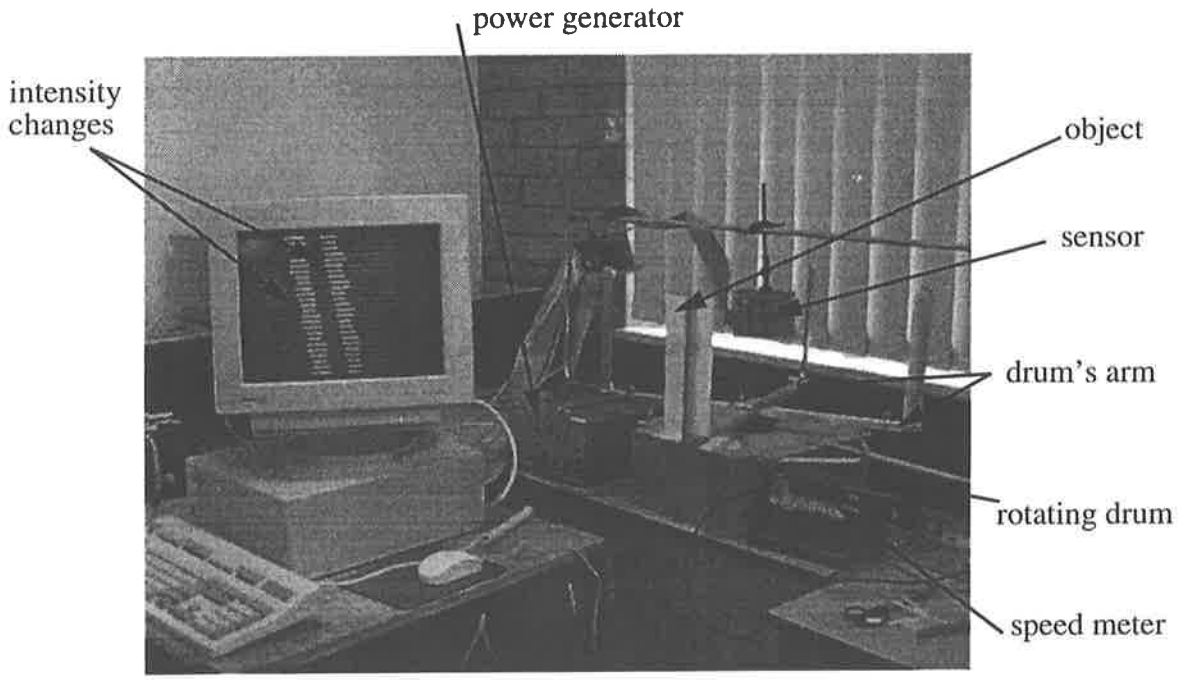


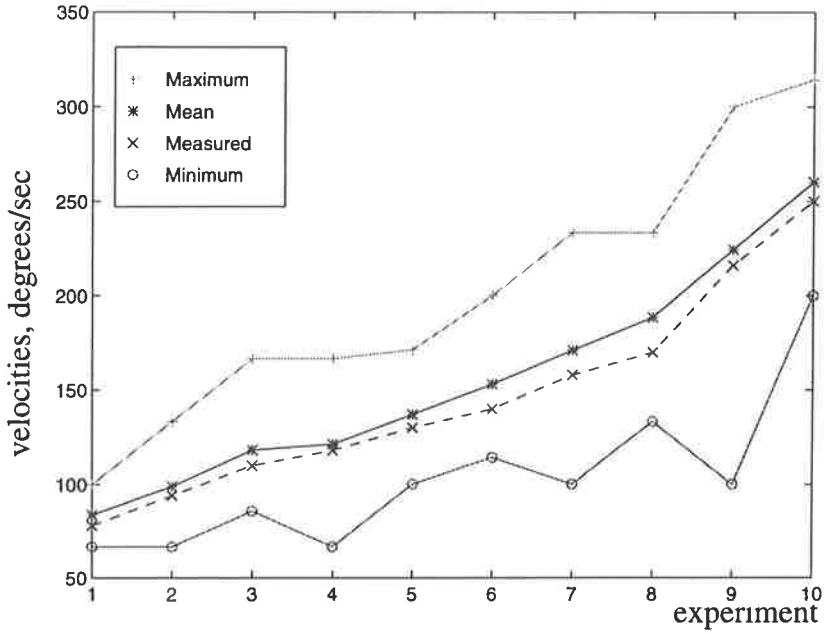
Fig. 8.17 Apparatus of the rotating drum experiment.

8.8.3.2 Results

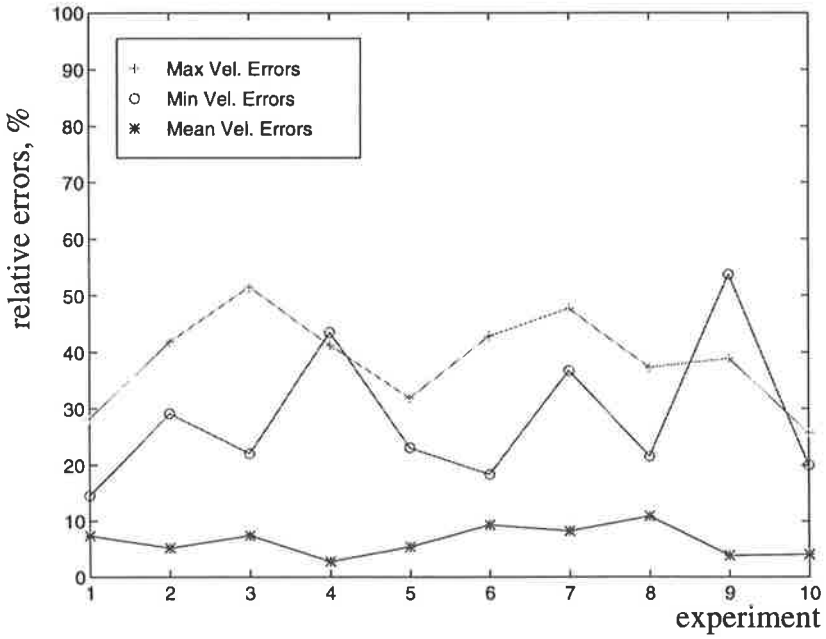
The output of the sensor consists of intensity changes which are recorded into files on the computer for a period of time. The velocity estimates are then computed from the recorded data using the backward tracking algorithm, which computes a distance D (channels or degrees) travelled by and the corresponding time steps n_s taken by the object between eight consecutive sampling instants. Hence, for a used sampling interval of 5 milliseconds, the estimated velocity is given by:

$$velocity = \frac{D}{n_s \times 0.005} \quad \text{degrees per second} \tag{8.1}$$

Ten experiments were carried out with ten different velocity settings by increasing the current applied to the step motor. The measured velocities were chosen arbitrarily as 78, 94, 110, 118, 130, 140, 158, 170, 216, and 250 degrees per second. Note that irregular increments between the measured velocities are due to the manual adjustment of the current-control knob. If the rotating velocity of the drum is either lower that the minimum setting of 78 degrees per second (i.e. 0.39 degrees per 5 milliseconds, or 0.39 channels per sampling interval) or higher that 250 degrees per second (i.e., 1.25 channels per sampling interval), templates induced by motion cannot distinguished from templates due to other factors such as noise. Hence, velocities cannot be estimated reliably.



a)



b)

Fig. 8.18 Experimental results of velocity estimation. a) Maximum, averaged, and minimum estimated velocities, and physically measured rotating velocities, versus the numbered experiment. b) Relative errors.

For each experiment, a large number of velocity estimates are computed. The mean velocity is the average value among the estimates for one experiment. Similarly, the maximum and minimum values are also selected as the highest and lowest values among the estimates.

The experimental results are plotted in Fig. 8.18, which shows maximum, average, and minimum estimated velocities, and physically measured rotating velocities for ten experiments; and relative errors of the estimates.

The average estimates are higher than the measured values by up to 10%, which is consistent with the results obtained in the simulation of Section 6.4. As explained in Chapter 6, these errors might be caused by the spatial quantization of the visual field into the detecting channels. However, as the errors are all positive, there might be other causes, such as misalignment of the apparatus, other than the spatial quantization. One possible reason that contributes to these positive errors is due to the interface between the lens and the chip. As the lens is glued on the chip, there exists an intermediate layer between the chip's surface and the lens's surface as shown in Fig. 8.19. This intermediate layer may cause the image to be out of focus. The size of images is consequently increased slightly compared to the images obtained from the ideal case of the lens touching the chip's surface as illustrated in Fig. 7.4. As a result, measured angular distances travelled by the object are longer, and hence estimated average velocities are higher as shown in Fig. 8.18a.

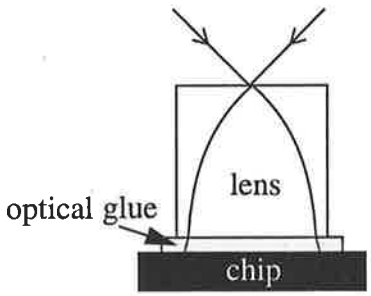


Fig. 8.19 The optical interface in practice.

On the other hand, the errors of the maximum or minimum estimates go occasionally up to 50%, which can be interpreted as follows. The errors are caused by raw velocity estimates in only short time periods, especially when only the first two occurrences of a DMST are available as the object just enters the visual field. In other words, the maximum and the minimum estimates are usually the values which are averaged over a period of less than eight sampling instants. Thus, the errors are high as they are inversely proportional to the time period (Appendix E).

8.9 Summary

In this chapter the design of the smart micro-sensor II is presented. The system is meant to provide a test-bed for analog circuits and for implementing the backward tracking algorithm. The constituting processors of sensor II are functionally more advanced than those of sensor I. However, unknown problems in the analog differentiator effectively prevent the constituting processors into a complete system.

Despite the aforementioned problems, experimental results are very encouraging, and confirm the following:

- Motion of moving edges are indicated by pairs of DMSTs (Section 4.6.2).
- The template and velocity estimation processors work with real-time template data. The simulation confirms the validity of the designs of the template and velocity estimation processors, and hence are ready to be mapped into a custom-designed VLSI chip in a future generation.
- Estimated average velocities using the backward tracking algorithm are obtained within 10% errors, which, it is assumed, are caused by the spatial quantization of the visual field into the detecting channels, and by the rough optical interface between the chip and the lens.

Chapter 9

Conclusions

9.1 Thesis Summary

This thesis has presented the principles of motion detection of insect vision and the hardware implementation of those principles, via the template model, creating smart microsensors I and II. In fact, the sensors' real-time operation shows that the principles of insect vision are physically realizable.

In Chapter 1, the research technologies and methodology were established. The research technologies are the principles of insect vision and the mixed analog-digital VLSI. The Marr methodology was adopted as a standard for this research, where the study of an information processing system should begin with a study of the computational theory followed by an analysis of the data representation and algorithms to process the data, and finish with a hardware design, which is an essential phase to demonstrate that the studied information processing system may be successful.

The first important discussion (Chapter 2) concerns evolutionary perspectives in the field of the visual motion computation. It is argued that the real-time application of existing techniques, which usually concentrate on the 3D representation of the environment, suffer from computational bottleneck and are usually not robust against noise. Generally, the existing techniques mathematically lead to the problem of minimizing an associated functional, which requires a computational cost using a numerical iterative method. These techniques are also sensitive to noise because of the needs for numerical differentiation. Furthermore, the 3D representation of an environment is not necessary in some applications in the sense that an intelligent agent, like a mobile robot, may not need to know the configuration of the environment in order to navigate around obstacles.

In contrast, biological vision systems, such as insect visual systems, are very simple, and do not have virtually unlimited resources in terms of processing power and storage capacity compared to digital computers used in solving visual tasks. However, insects can see better than machines do, and indeed, offer a number of lessons for machine vision such as task-oriented vision, motion detection, parallel processing, and so on. Insects rely on motion

information to estimate their positions relative to nearby objects. Hence, motion detection is essential in insect vision.

Ample evidence from different disciplines studying biological visual motion shows that the motion detection mechanism should possess an asymmetrically nonlinear interaction between two input channels (Chapter 3). These findings in biological vision systems have helped researchers to formulate different models for motion detection such as the correlation model and the shunting lateral inhibition model (Chapter3).

In almost the same context, the template model was proposed to explain the phenomenon of jumped motion found in the insect visual system. However, it was observed that the development of the template model is not complete due to the lack of a logical interpretation on the formulation of templates, and hence their physical meanings. The template model was then shown to be qualified as a motion detection model according to the principles of motion detection (Chapter 4). Its directional selectivity was formulated, and then was used to identify eight templates sensitive to coherent motion, which are named directionally motion-sensitive templates (Chapter 4).

The benefits of the template model were investigated in Chapter 5, where the directionally motion-sensitive templates together with the position conjugate templates could be employed as features to represent motion trajectories in the spatio-temporal domain. Two different schemes of representing motion trajectories were proposed. Scheme 1 is the staircase representation and scheme 2 is the feature representation. The two schemes offer two advantages. Firstly, the two schemes use only information related to coherent motion. In other words, an extraction of useful information is achieved. Hence, a data reduction is obtained. Secondly, the two schemes open up ideas for proposing new tracking techniques for velocity estimation, which are the focus of Chapter 6. The basics of velocity estimation and three different tracking techniques, stairs-step tracking, forward tracking, and backward tracking algorithms, which are used to obtain velocity information, were discussed. Comparatively, the backward tracking algorithm is the most promising technique, which was chosen for the implementation (Chapter 8).

The purposes of Chapters 7 and 8 were to demonstrate two different VLSI designs of the smart micro-sensors I & II respectively. Sensor I was meant as a concept demonstrator, while sensor II was aimed at a test facilitator. Sensor I could detect only dark-to-bright moving edges. This achievement nonetheless indicates that the principles of motion detection of insect vision can be reproduced into hardware using advanced VLSI technology. Sensor II was not completely integrated because of some yet unresolved problems occurring in the analog differentiator. However, the experimental results confirm the claims on responses of the DMSTs and on the feature representation (Chapters 4, 5 & 8).

9.2 Ongoing Works and Investigations

As mentioned in Chapters 7 & 8, the low-frequency analog differentiator is still the most critical issue in the design of these smart micro-sensors. However, different configurations of the low-frequency analog differentiator have been designed and fabricated as test patterns (Moini private communication). Other alternatives could be considered to avoid this critical problem, for example, digital filters are currently investigated (group private communication).

In senor II, the digital processors, i.e., the template and velocity estimation processors, are implemented using Altera EPLDs, the digital processors should be combined with the analog front-end to form a monolithic compact smart micro-sensor once the problem of the analog differentiator is solved.

The distortion of the GRIN lens creates a non-uniform distribution of the receptive fields radially (Appendix A). So, it is desirable to have a real compound-eye-like smart sensor, i.e., each photodetector has its own tiny lens, and hence the photodetectors have, at least in principle, the same receptive field. This idea has been recently proved to be plausible (Ogata et al.). Furthermore, a new technology, called *binary optics*, could produce micro-lenses by etching the surfaces of optical material, using techniques similar to those employed in the fabrication process of integrated circuits. The advance of this technique is to make micro-lenses whose specifications are limited only by current VLSI technology. Thus, the binary optics can be a promising alternative in designing the optical interface of these smart micro-sensors.

Appendix A

Optical Interface

No matter how greatly the apparatus for seeing may vary from one kind of animal to another, vision is always the same in one fundamental respect: the process begins with light entering the eye and bringing with it the information it has picked up in touching or passing through the objects in its paths.

[Mueller & Rudolph, 1972]

A.1 Introduction

Any vision system principally consists of two parts: optical interface and sensory signal processor. Thus, a vision system does not definitely work without either part. In other words, the optical interface, which is used to focus light into the sensory signal processing part, is as important as the sensory signal processor in a vision system. The optical interface serves as a mapping device by which a 3D world is projected onto a 2D image plane basically formed by light-sensitive cells. In human vision, these cells are photoreceptors lying on the back wall of the eye, called the *retina*, while in machine vision, these cells are photodetectors, which convert light energy into electrical signals. It depends on individual vision system, the optical interface is differently constructed, for example, it is a single lens in the human's eye but is a system of tiny lenses in the insect's eye (Nilsson 1989).

In designing machine vision systems, the optical interface should be considered carefully because the optical resolution of the systems is determined partly by the characteristics of the lens chosen for the implementation. That is, based on the optical properties of the lens, the size of photodetectors is determined to produce a dimensionally compatible sensory signal processing counterpart.

This appendix concerns the optical interface of the smart micro-sensors I & II, and is organised as follows. Section A.2 introduces a panoramic vision system proposed in this research. Section A.3 gives some requirements for the optical interface of this specific system. Section A.4 provides the optical characteristics of a gradient index lens which is chosen for the implementation. Section A.5 expresses the optical resolution of the smart micro-sensors, while Section A.6 presents the physical layout of the photodetectors. The appendix ends with a summary.

A.2 Panoramic Machine Vision

Panoramic machine vision is desirable for mobile robots to detect all obstacles around them without the need for rotation. There are a number of robot vision systems which have this panoramic feature (Pichon et al. 1989; Zheng & Tsuji 1992; Yagi el al 1994). In nature, panoramic vision did exist very early, for instance, the earliest ancestors of the anthropods, 400 million years ago, had the 360^{o} vision in the horizontal plane, whose major functions were for navigation during swimming and stabilizing against rotation (Horridge 1992).

A panoramic machine vision system for motion detection and velocity measurement, which is proposed in this research, has a visual field encompassing 360^{o} in a horizontal sense. This means that the system detects moving objects on a fixed horizontal plane. Ideally, the photodetectors of the system may lie on the wall of a cylinder; each of the photodetectors looks in a different direction. However, this ideal structure cannot be implemented in the present VLSI technology, and would also raise some difficulties in designing an optical interface for such a system. An alternative is to partition the 360^{o} visual field into six 60^{o} sub-fields, each of which is equipped with one smart VLSI micro-sensor. Fig. A.1 shows one possibility of a very simple physical arrangement of the panoramic visual system, which consists of 6 identical smart VLSI micro-sensors placed on side walls of a hexagonal-base cylinder.

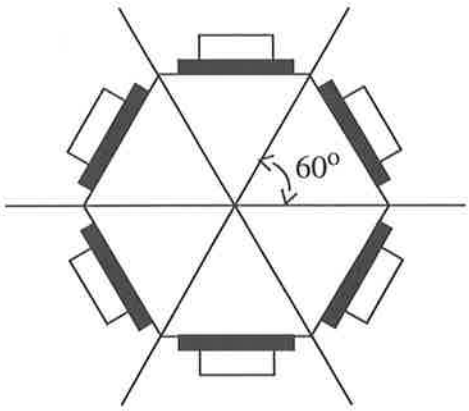


Fig. A.1 Top view of a physical arrangement of a paranomic vision system consisting of 6 identical smart VLSI micro-sensors.

A.3 Requirements for An Optical Interface

Based upon the architecture of the proposed panoramic vision system, the first requirement of the optical interface of the constituting smart micro-sensors is that the accept-

ance of any optical device is at least 60^{o}. The optical device should also have dimensions compatible to the size of the VLSI sensor, which is in the order of 1 to 2 mm. If the optical device is large, the sensor's size is also large, and hence the VLSI real estate is more expensive in terms of the fabrication cost. Furthermore, the optical device should be easily assembled onto the sensor. Devices which satisfy all the requirements are *gradient index* (GRIN) lenses. These lenses may have quite small sizes suitable for VLSI devices and have reasonably large acceptance angles. In fact, tiny gradient index lenses naturally appear in the insect visual system (Kunze & Hausen 1971) and they can now be implemented for artificial compound eyes (Ogata et al. 1994).

A.4 Gradient Index Lens

In conventional optical devices, the index of refraction within each device is considered to be homogeneous, that is, the refractive index is constant within the material. Within advanced technologies in optics, it, however, is possible to manufacture lenses whose refractive index varies continuously within the material. Such lenses are called *gradient index* lenses. The GRIN lenses are a fairly new development in the field of optics because the technology that is able to produce them has only been available for a short time. In the commercial GRIN lenses (available only since 1976), the index varies radially, decreasing parabolically out from the central axis, called GRIN-rod lenses (Hecht 1990).

A.4.1 Basic Imaging Properties of GRIN-Rod Lenses

A GRIN-rod lens consists of a cylinder of dielectric material with a radial refractive index distribution described by Eq. (A.1) (Melles Griot)

$$N(r) = n(0)\left(1 - \frac{Cr^2}{2}\right) \tag{A.1}$$

where $N(r)$ is the index of refraction of the GRIN-rod lens at a radial distance r from the optical axis, while $n(0)$ is the refractive index at the optical axis; C is a constant.

As a result of this parabolic index distribution, a ray incident on the front surface follows a sinusoidal path along the rod lens as shown in Fig. A.2. The pitch of the lens is defined as the period of the sinusoidal path. The maximum incident angle in which the lens forms images is called the acceptance angle of the lens. The length of the lens determines imaging characteristics of the lens, for instance, focal plane.

Based on the requirements for the optical interface in Section A.3, the lens which is chosen for the implementation is a quarter-pitch GRIN-rod lens, which forms an inverted

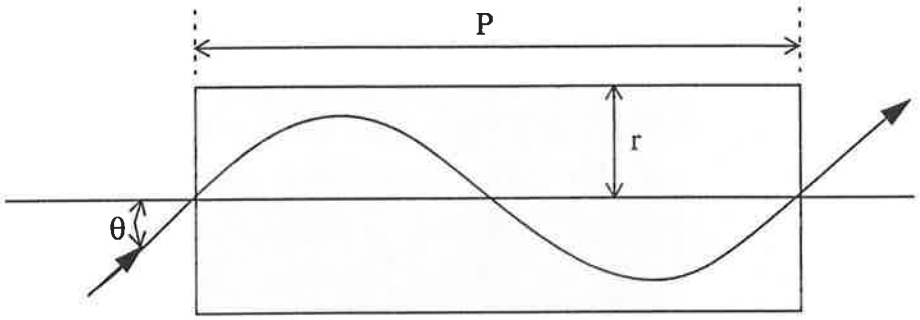


Fig. A.2 Sinusoidal ray path in a GRIN lens. P is the pitch of the lens.

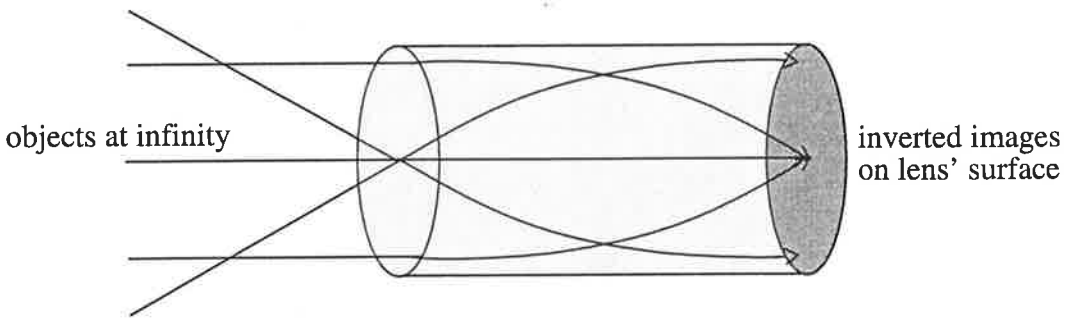


Fig. A.3 Ray trace through a Selfoc quarter-pitch GRIN lens.

Fractional pitch	*P*	0.25
Acceptance angle	α	72°
Length	*L*	3.7 mm
Radius	*R*	0.9 mm
Design wavelength	λ	623.8 nm
Numerical aperture	*A*	0.6
On-axis refractive index	*n(0)*	1.658
Constant	*C*	0.183 mm^{-1}

Table A.1 Specifications of the quarter pitch GRIN lens (Melles Griot catalogue).

and real image of an object at infinity on the opposite end of the lens. Fig. A.3 shows what the ray trace through such a lens looks like. The physical dimensions and material properties of the quarter-pitch GRIN lens are listed out in Table A.1.

A.4.2 Aberrations of GRIN-Rod Lenses

In practice, all optical devices are subjected not only to diffraction but also aberrations. The diffraction is due to the fact that the lens is not able to focus all the wavefronts propagating from the object, but just a fraction of them. This phenomenon results in a small spot when a lens forms an image of a point source. The smaller the diameter of the lens, the smaller fraction of the wavefronts, and the larger the diffraction pattern that results in. Other imaging errors may also be caused by different aberrations such as spherical, chromatic, coma, astigmatism, and so on, which are the names given to various departures from the ideal performance of a lens (Hecht 1990). The parabolic refractive index profile of a GRIN lens however makes the analytical calculation of the parameters of the aberrations extremely complex, and indeed, the aberrations of GRIN lenses are still the subject of research in the optics field (Tomlinson 1980; Sakamoto 1984; Marchand 1986).

The blurring effect due to the aberrations, that is, a point source is mapped onto a blurring spot instead of a point-sharp image, is of interest in determining the optical resolution of the smart micro-sensors, and hence is investigated in this study.

In fact, the GRIN lens has a negligible blurring effect due to the diffraction compared to that due to the aberrations. For a numerical aperture greater than 0.5, the spot diameter due to the diffraction is smaller than 1 μm, while that due to the aberrations is of the order of 5 μm (Iga 1980). A general formula, which estimates a diameter of a burring spot due to the spherical aberration, is given in Eq. (A.2) (Tomlinson 1980).

$$d = 0.007 \times \frac{L}{32} \times \left(\frac{A}{0.2} \right)^3 \qquad \text{(A.2)}$$

Using the given data of the quarter-pitch GRIN lens from manufactured by Selfoc (Melles Griot), the diameter of a blurring spot formed from a point source due to the aberrations is computed as 22 μm.

A.5 Optical Resolution

The optical resolution of a sensor is determined by a trade-off between the characteristics of the lens and the constraints of the technology used to design the processing part, which is CMOS VLSI in this project, and the application of the sensor. Essentially, the an-

gular resolution of motion detection systems, either biological or machine, is in the order of 1 to 3 degrees (Franceschini et al. 1992; Horridge 1992; Westheimer & Wehrhahn 1994). Learning from these systems, it is suggested that the angular resolution of the proposed panoramic system should be 1 degree, that is, one photodetector has a receptive field of 1^{o} for a better optical resolution. Thus, each smart micro-sensor should have a row of 60 photodetectors.

With the selected quarter-pitch GRIN lens, a 1^{o} receptive field corresponds to 25 μm on average, (i.e., 1.8 mm ÷ 72^{o}). As the spherical aberration of the lens produces a blurring spot of 22 μm (i.e., corresponding to 0.89^{o} on average), the effective receptive field corresponding to 25 μm is approximately 1.89^{o}, and has a Gaussian shape, and hence creating an overlap between the receptive fields of adjacent photodetectors as shown in Fig. A.4.

In nature, the overlap between the adjacent receptive fields is essential in the insect's compound eyes, which helps to eliminate the aliasing problem associated with the sampling lattice of the tiny eyes (Borst & Egelhaaf 1989) and to increase the sensitivity to a diffuse source (Horridge 1992). Furthermore, an object moving across a row of photoreceptors can be distinguished continuously as the same object.

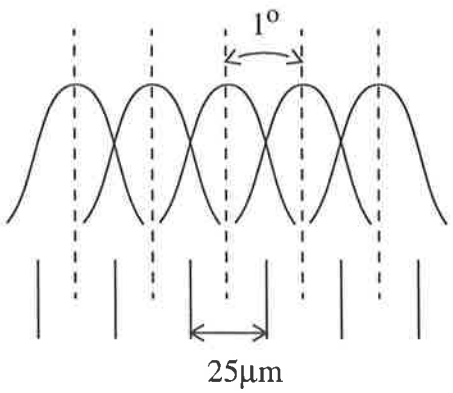


Fig. A.4 Overlapping receptive fields caused by the aberration effect.

A.6 Physical Layout

Although the horizontal maximum pitch of a photodetector is 25 μm, in practice the size of the photodetector is smaller because there should be some space between adjacent photodetectors as required by the design rules of a VLSI technology process. This means that the minimum spacing between adjacent photodetectors is set by the design rules. Using Orbit's 2 μm CMOS technology, this minimum spacing is 16 μm for a safe design. In con-

trast, the maximum spacing is set by the size of the blurring spot, which is 22 μm, in order to cover the whole visual field, i.e., there is no gap between adjacent receptive fields.

Thus, in order to ensure the overlap between adjacent receptive fields, a conservative choice for the spacing between adjacent photodetectors is 18 μm, and hence the photodetector's width is 7 μm corresponding to a receptive field of 1.16° (i.e., (7+22) ÷ 25). In the vertical direction, the photodetectors are designed with a receptive field of 2° on average to compensate for any misalignment errors which may occur due to the set up being not exactly horizontal or the ground being sloped slightly. Thus, the photodetector's length is 50 μm. Fig. A.5 illustrates the physical layout of the photodetectors of the smart micro-sensors.

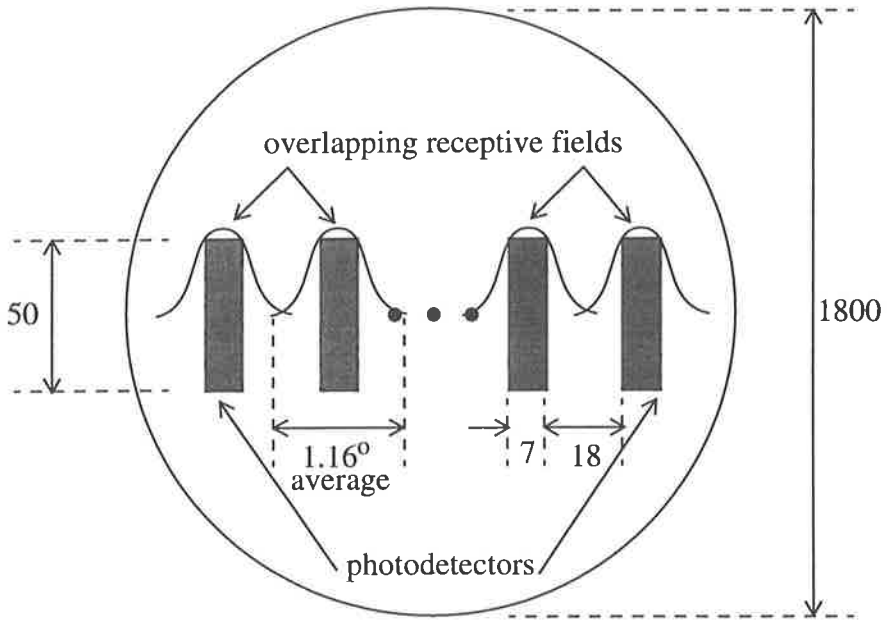


Fig. A.5 A physical layout of photodetectors underneath the surface of a quarter-pitch GRIN lens, dimensions in μm.

It is worth noting that the GRIN lens is also subjected to the Barrol distortion (Hecht 1990), which creates a non-uniform distribution of the receptive fields. Experiments carried out by Andrew Blanksby (private communication) show that 1° maps to approximately $28\mu m$ near the lens centre, and $18\mu m$ to towards the lens boundary. Consequently, the designed photodetectors have the receptive fields of 1.04° (i.e., (7+22) ÷ 28) and 1.61° (i.e., (7+22) ÷ 18) respectively. Thus, the row of the photodetectors still covers the whole visual field. However, the real-time experiments of the smart micro-sensor II reveals that the distortion of the GRIN lens does not substantially affect the errors of estimated average velocities of moving objects (Chapter 8).

A.7 Summary

In this appendix we have described the physical architecture of the proposed panoramic vision system, which may be constituted from 6 identical smart VLSI micro-sensors. We have also discussed the requirements for the optical interface of the smart micro sensors. Based upon these requirements, the Selfoc quarter-pitch GRIN-rod lens is chosen for the implementation. This lens suffers from the diffraction as well as Barrol distortion effects, which are, however, negligible in this specific application. In contrast, the aberration effect is useful in terms of creating an overlap between the receptive fields of adjacent photodetectors, which is essential in spatially sampling vision systems.

Appendix B

Software Simulation

A template-based detector is constructed as a row of detecting channels which have overlapping receptive fields. Each channel detects changes in light intensity by using a bandpass filter which has a bandwidth of 100Hz. A threshold operation is then applied to further eliminate noise contamination in filtered signals, which are then quantized into one of the three primitive states: *increase*, *decrease*, or *no-change*. Fig. B.1a illustrates a spatio-temporal profile of intensity signals obtained from a bar moving in front of the sensor, while Fig. B.1b shows a corresponding spatio-temporal profile of temporally bandpass-filtered intensity signals with two thresholding levels. Signals which are above the positive threshold are considered as increase in intensity represented by ('i'), while signals which are below the negative threshold are considered as decrease in intensity represented by ('d'). Any signals which lie between these two levels are classified as no-change in intensity displayed by dots ('.'). The thresholded signals are then sampled at 200 Hz, yielding data in the form of changes in intensity as shown in Fig. B.2a. Templates are then formed by combining the intensity change states between two adjacent spatial channels at two consecutive sampling instants as shown in Fig. B.2b in the format of conjugate pairs, i.e., showing both DMSTs and PCTs. Similarly, Fig. B.2c shows only DMSTs. The simulation is done using MATLAB and C programming language.

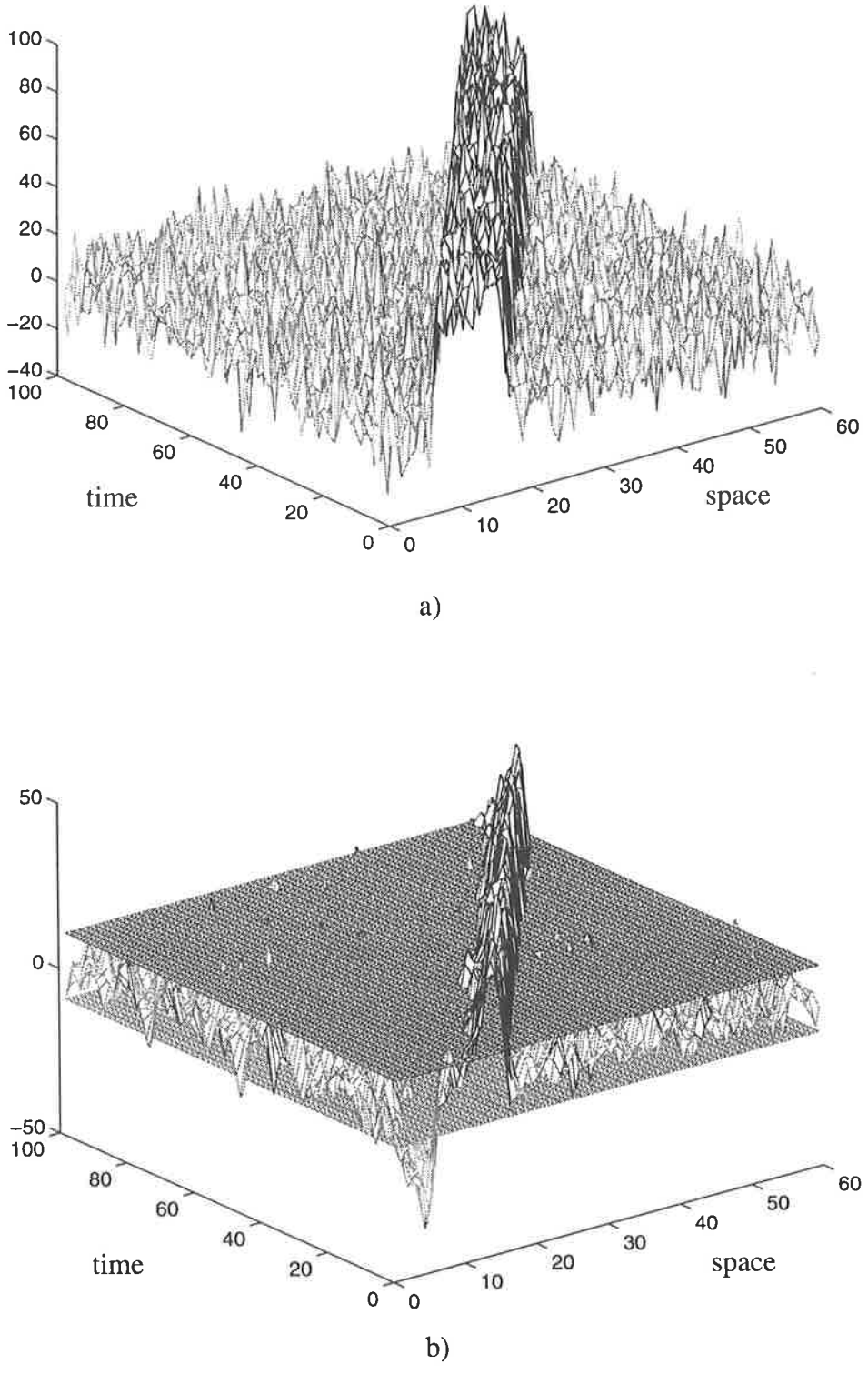


Fig. B.1 Spatio-temporal profiles of a) intensity signals and b) intensity change signals obtained from a bar moving in front of the sensor. Numbers on the vertical axis represent arbitrary units.

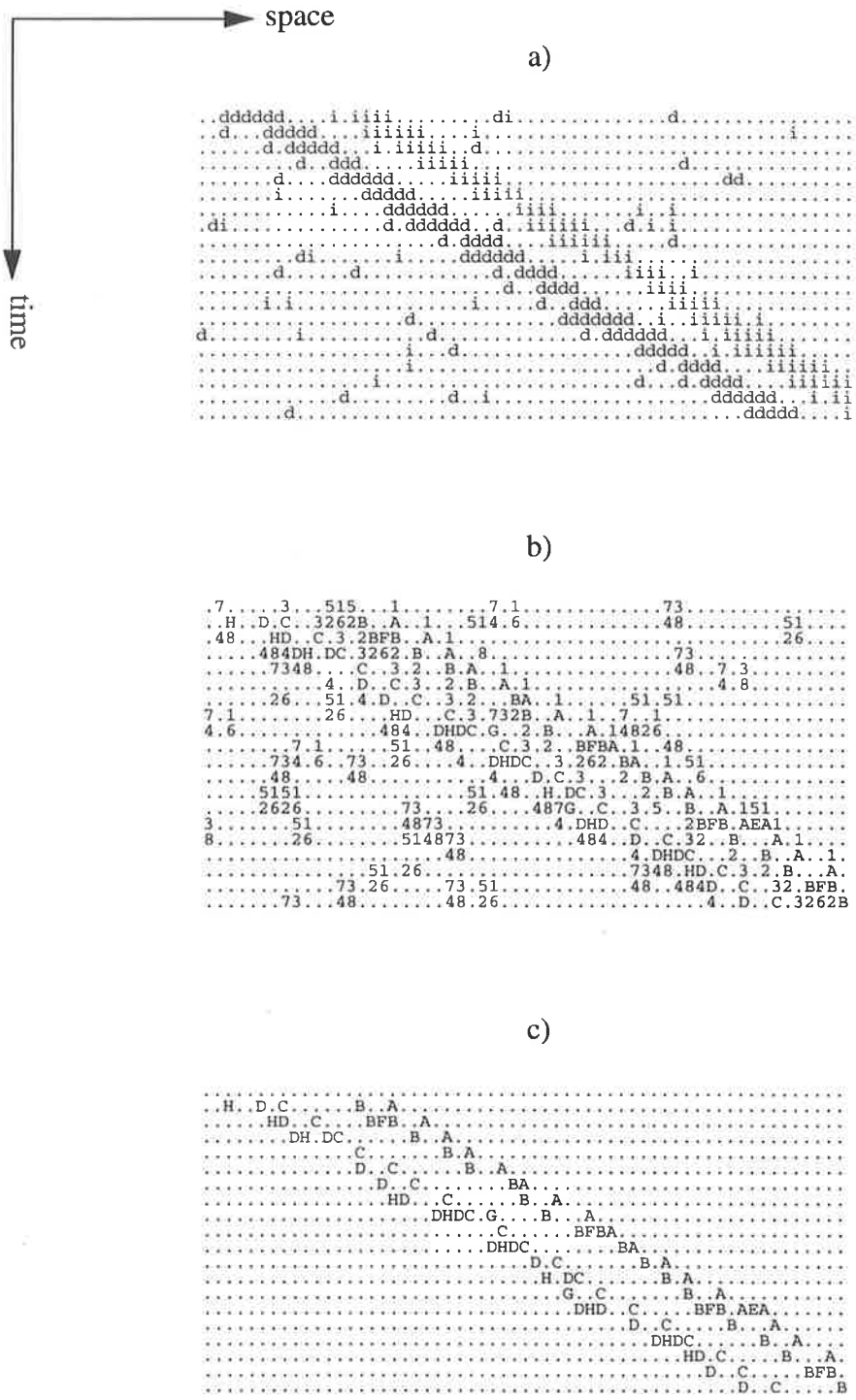


Fig. B.2 Examples of simulation results in the format of: a) changes in intensity. b) conjugate pairs. c) DMSTs only.

Appendix C

Responses of Directionally Motion-Sensitive and Position Conjugate Templates

Appendix C presents responses of directionally motion-sensitive templates (DMSTs) and position conjugate templates (PCTs) at different sampling frequencies and at different thresholding levels.

The object used in the simulation is a white bar with respect to a dark and uniform background. The bar non-uniformly moves from right to left and back for one and half cycles. Although there are eight conjugate pairs (DMST, PTC) as discussed in Section 5.5.1, responses of only two pairs, (A,1) and (E,5) indicating motion to the right and to the left respectively, are considered for the purpose of demonstration. The responses of the DMSTs and PCTs are numbers of detected DMSTs and PCTs respectively, which are counted at five different sampling frequencies of 100, 200, 250, 333, and 500 Hz †, and at 6 different thresholding levels of 10, 12, 14, 16, 18, and 20% of the maximum value of intensity change. Fig. C.1 depicts the responses of (A,1), while Fig. C.2 depicts the responses of (E,5).

Fig. C.1a and Fig. C.2a show that as the frequencies are increased above 200Hz, the responses of DMSTs are saturated. If the sampling frequency is below 200 Hz, the aliasing problem occurs since the bandwidth of the change-detecting circuit is 100 Hz, and hence some DMSTs are missing. The responses of PCTs are linearly increased with an increase of the sampling frequencies at low thresholding levels, while the responses of PCTs are compatible to those of DMSTs at high thresholding levels, i.e., the response of PCTs are inversely proportional to the thresholding level.

Fig. C.1b and Fig. C.2b show that the responses of DMSTs do not vary much when the thresholding level is increased. In contrast, the responses of PCTs are rapidly reduced to the responses of their DMST counterparts as the thresholding level is increased. In short, the thresholding level has a strong effect on the responses of the PCTs and while has a little effect (or no effect) on the responses of the DMSTs, which can be explained in the following

†. Five different frequencies are not linearly selected in order to facilitate numerical simulations using MATLAB. Intensity signals which are initially sampled at 1000 Hz are filtered and thresholded. Then thresholded signals are re-sampled at one of the selected frequencies before being combined into templates. The five selected frequencies are the results of dividing 1000 Hz by factors of 10, 5, 4, 3, and 2 (i.e., 100, 200, 250, 333.3, and 500 Hz).

discussion.

Recall from Section 5.5.1 that the PCTs have a single intensity-change entry which in turn makes the PCTs being weak responses because such a single intensity-change entry can be easily generated by any random intensity fluctuations (i.e., noise) over the background. To ensure that the PCTs are induced by motion only, such random intensity fluctuations should be removed so that a noise-free background can be obtained.

In the template model (Chapter 4), the thresholding operation is employed for the purpose of removing noise. That is, any fluctuations which have the magnitude less than the thresholding level are considered as noise and such fluctuations are suppressed. Therefore, as the thresholding level is decreased, the level of noise contamination in thresholded signals is high. Hence, the responses of PCTs are high. In contrast, as the thresholding level is increased, the level of noise contamination in thresholded signals is low. Hence, the responses of PCTs are low. In other words, the PCTs are sensitive to noise.

On the other hand, the thresholding level does not strongly affect the responses of the DMSTs because, compared to occurrences of the PCTs, random fluctuations hardly generate a group of either three intensity-increase entries or three intensity-decrease entries unless the thresholding level is set very low.

Finally, if the thresholding level is set to sufficiently high so that all noisy intensity fluctuations are completely eliminated, then intensity changes are induced by motion only. Consequently, the DMSTs and PCTs are caused by motion only, and hence their responses are expected to be similar because the DMSTs and PCTs occur simultaneously in pairs. However, movements of objects which have low contrast may not be detected if the thresholding level is set too high. So, the thresholding level should be chosen as a trade-off between the responses of the DMSTs and of their conjugate counterparts.

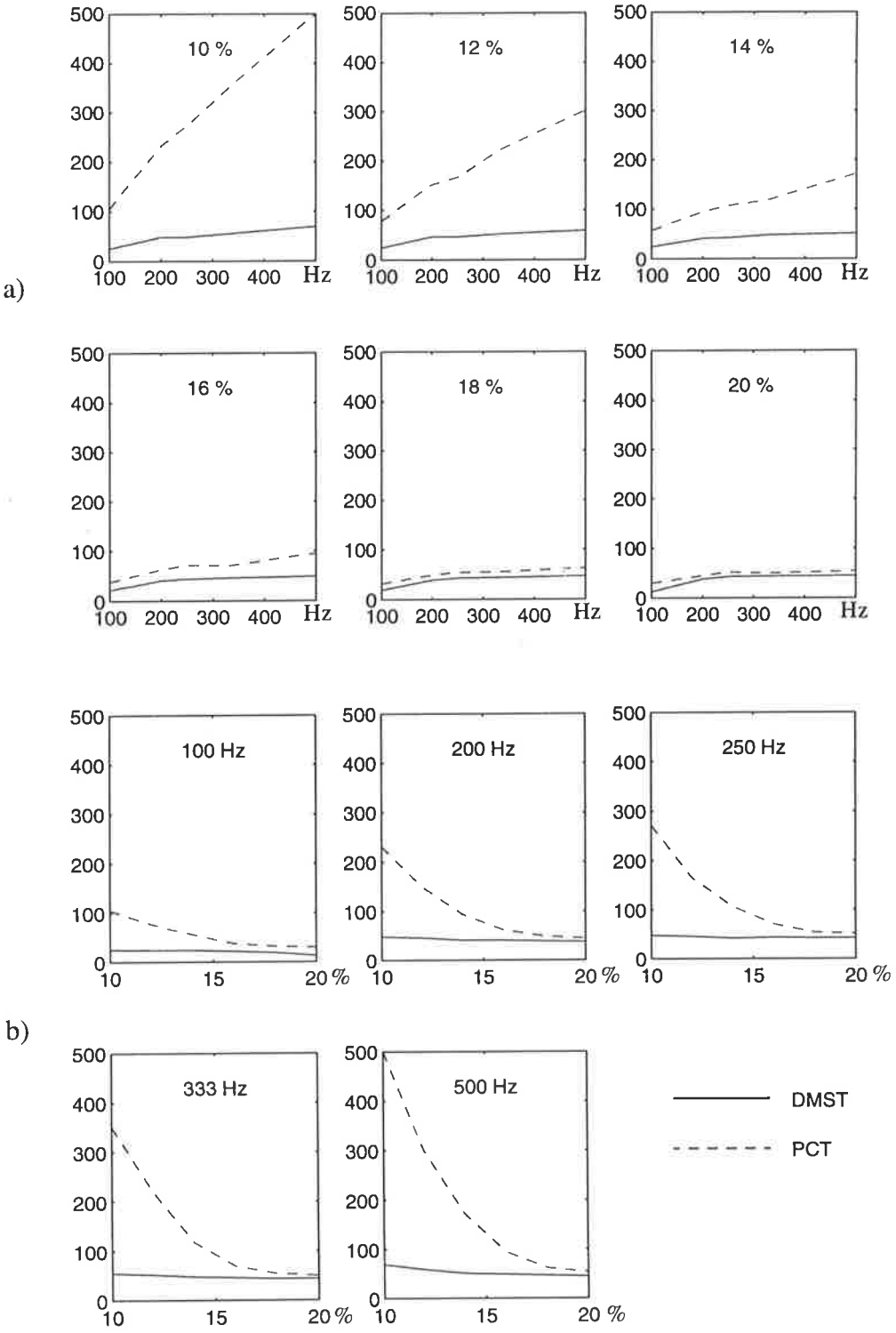


Fig. C.1 Numbers (vertical axes) of the detected DMST 'A' (solid curves) and of the detected PCT '1' (dashed curves) versus: a) sampling frequencies (Hz) at six different thresholding settings of 10, 12, 14, 16, 18, and 20% of the maximum intensity change, and b) thresholding settings (%) at five different frequencies of 100, 200, 250, 333, and 500 Hz.

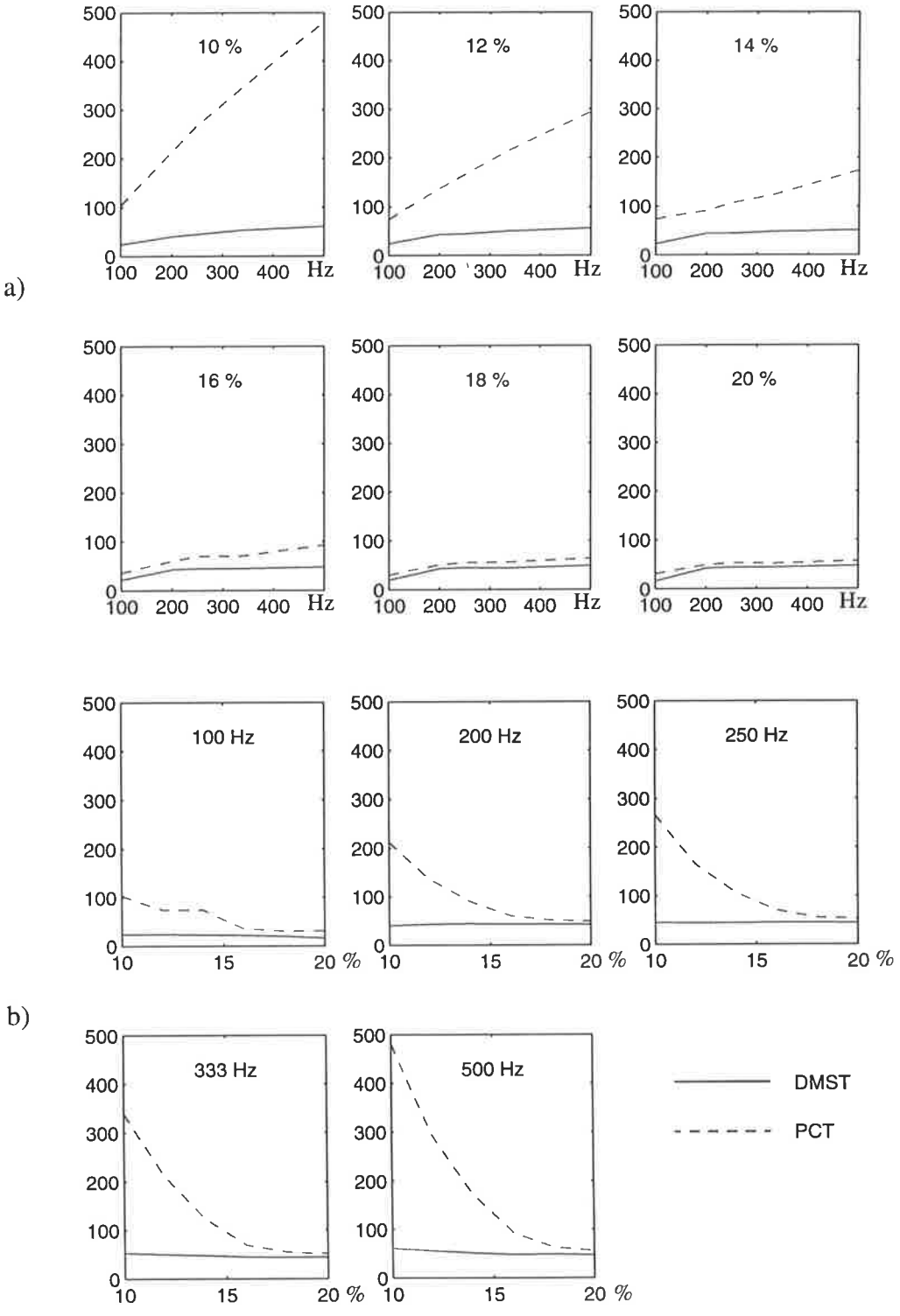


Fig. C.2 Numbers (vertical axes) of the detected DMST 'E' (solid curves) and of the detected PCT '5' (dashed curves) versus: a) sampling frequencies (Hz) at six different thresholding settings of 10, 12, 14, 16, 18, and 20% of the maximum intensity change, and b) thresholding settings (%) at five different frequencies of 100, 200, 250, 333, and 500 Hz.

Appendix D

Hough Transform for Velocity Estimation

D.1 Fundamentals of Hough Transform

Let us consider a straight line in the XOY plane as shown in Fig. D.1. This straight line is specified by

$$y = x\tan\beta + c \tag{D.1}$$

or

$$y\cos\beta = x\sin\beta + c\cos\beta \tag{D.2}$$

where β is the angle subtended between the x-axis and the line, and c is the y-intercept of the line.

By using trigonometric formulae and definitions of sine and cosine, Eq. (D.2) can be expressed as

$$\rho = x\cos\alpha + y\sin\alpha \tag{D.3}$$

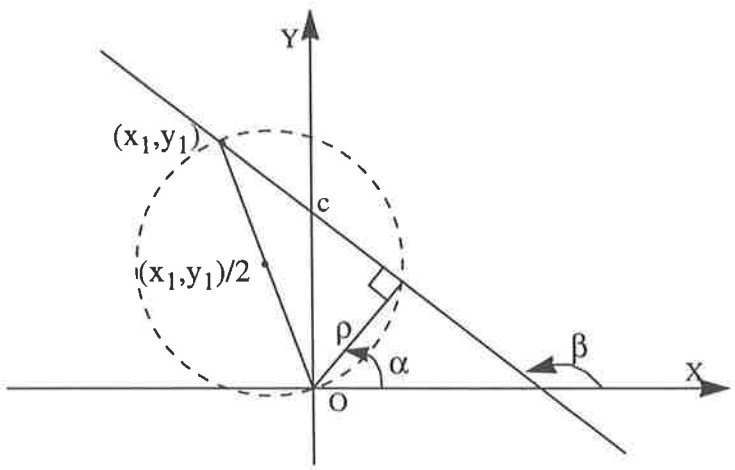


Fig. D.1 Relation between x, y, c, β, α, and ρ of a straight line.

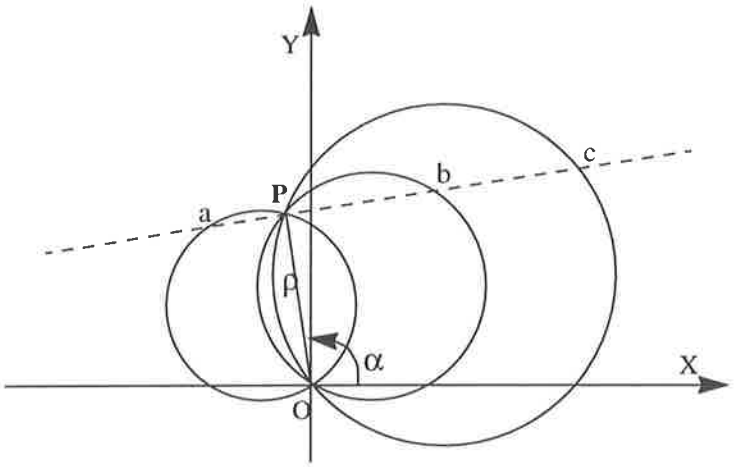


Fig. D.2 Determination of a straight line passing through three points a, b, and c.

where the quantities ρ and α are the respective length and the angle with respect to the x-axis, of the position vector at the foot of the perpendicular to the straight line as illustrated in Fig. D.1. Eq. (D.2) constitutes a two-parameter family. If a parameterization for the family is fixed, then an arbitrary straight line can be represented by a single point in the parameter space (Duda & Hart 1972).

The locus of the feet of the perpendiculars to the set of the straight lines which pass through (x_1,y_1) is thus a circle with the centre at $(x_1,y_1)/2$ (Sklansky 1978). In the parameter space, each circular locus is equivalent to a sinusoidal curve represented by Eq. (D.3) with a fixed point (x_1,y_1).

Geometrically, a straight line passing through a, b, and c, as illustrated in Fig. D.2, is determined by the intersection point, P, of the three individual loci of these points (Sklansky 1978). This intersection is also determined in the parameter space as the intersection of the three corresponding sinusoidal curves. Thus, the problem of detecting collinear points is now converted to the problem of finding concurrent curves.

Conventionally, n points in an image can be mapped into the n corresponding sinusoidal curves in the parameter space, hence there will be $n\,(n-1)\,/2$ points of intersection between all pairs of curves. The intersection of these n sinusoidal curves, corresponding to n exactly collinear points, can be found by locating coincident points of intersection in the parameter space. Unfortunately, this approach creates a computational problem since the computing load grows quadratically with the number of points (Duda & Hart 1972).

Following Hough's basic proposal, the parameter space can be quantized into a quadruple grid with constraints on α and ρ of $0 \leq \alpha < \pi$ and $-R \leq \rho \leq R$, where R is the size of a 2D image of interest. The quantized region is treated as a 2D array of accumulating counters (cells). For each fixed point in the x-y plane, the corresponding sinusoidal curve is entered in the array by incrementing the count in each cell along the curve. After all points have been converted into the parameter space, the array is inspected to find cells having high counts. If the count in a cell is k, then precisely k points in an image lie along the line whose parameter is (α_i, ρ_i) (Duda and Hart 1972). The accuracy of this technique depends upon the quantization of α and ρ. The smaller the size of the quadruple grid, the better the accuracy is. However, the computation task is greater. Thus the quantization is chosen as a trade-off between the desired accuracy and the computational requirement.

D.2 Velocity Estimation using Hough Transform

The Hough transform is a powerful tool which was initially developed for detecting lines in images (Duda & Hart 1972; Brown 1984; Illingworths & Kittler 1988; Risse 1989). In order to estimate velocities of moving objects sensed by a template detector, this transform can be applied to detect interpolated lines passing through target templates such as DMSTs in a spatio-temporal image. The slopes of these lines are hence approximated as the angular velocities of moving objects.

In spite of the Hough transform's good recognition ability, its greatest disadvantages are a high computational cost and a requirement for large storage. To bypass the computational cost associated with the Hough transform when applied to the problem of velocity estimation with template data, the Hough transform was implemented slightly different to that described in the previous section. The procedure, which was used by Sobey & Horridge (1990), is summarised as follows:

> Velocities are roughly quantized into 16 levels which correspond to16 possible slopes as defined in Fig. D.3, where the minimum velocity is 4 channels per 16 time steps (i.e., 15 sampling intervals), and the maximum is 36 channels per 16 time steps. At the sampling instant, the channels are scanned across, and at each channel numbers of the target templates occurring on each of the 16 pre-defined slopes are counted extending back 15 previous time instants. There are such 16 numbers at each channel, which are stored into 16 counters. Thus, a 2D array of accumulating counters is needed to store these counted numbers. In this way, the spatio-temporal image of templates would be directly mapped into the space-velocity domain.

The technique was implemented in software without any consideration of its possible hardware architecture. Although the technique can be implemented in hardware (Hanahara et al. 1988), it may have a considerably complex architecture, since it may require at least two 2D storage arrays, one for template data, and one for velocity data in the transform domain. It also requires a look-up table for the transformation of data between the two domains. This table is necessary for mapping a template into different velocity levels since there may be a set of different slopes passing through one location of a target template. Furthermore, this technique is time consuming since it does not use information on the direction of motion, or the edge polarity provided by the templates. It searches globally for target templates in the image.

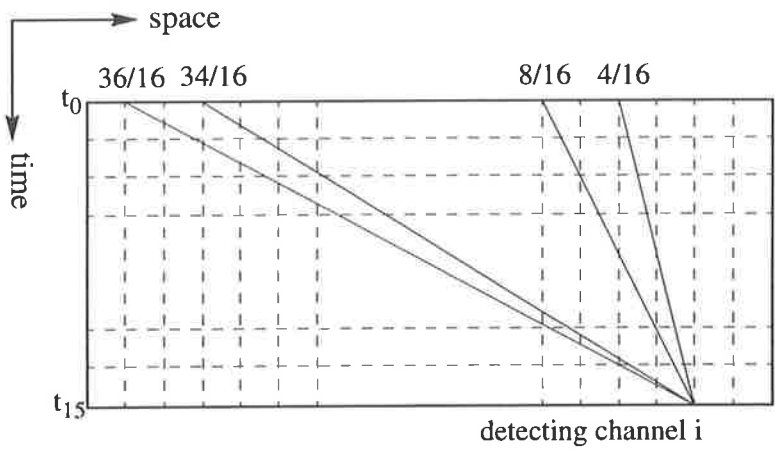


Fig. D.3 The modified Hough technique applied in a spatio-temporal image of templates. At each position (channel) in the last row, t_{15} (current time), a number of target templates occurring on each of 16 possibly quantized slopes is counted. The slope which has the highest number of the target templates would be chosen as an actual velocity among the 16 quantized slopes.

Appendix E

Quantization Errors

As the template model is involved with the spatial quantization of detecting channels, it is subjected to quantization errors, which may contribute to errors of estimated velocities of moving objects.

The quantizer operation on the channels $c(n)$ is denoted as $Q[c(n)]$ and let $c_q(n)$ denote the row of quantized channels. Hence,

$$c_q(n) = Q[c(n)] \tag{E.1}$$

Then the quantization error is a sequence $e_q(n)$ defined as the difference between the quantized value and the actual spatial value. Thus,

$$e_q(n) = c_q(n) - c(n) \tag{E.2}$$

This spatial quantized error $e_q(n)$ is limited to the range of $-w/2$ to $w/2$, where w is the receptive field subtended by a detecting channel, that is,

$$-\frac{w}{2} \le e_q(n) \le \frac{w}{2} \tag{E.3}$$

Subsequently, an estimated velocity, v_{es}, of an object moving from channel n to k for a time taken T, is

$$v_{es} = \frac{c_q(k)}{T} - \frac{c_q(n)}{T} \tag{E.4}$$

or

$$v_{es} = \frac{c(k) - c(n)}{T} - \frac{e_q(k) - e_q(n)}{T} \tag{E.5}$$

The first term in the right hand side of Eq. (E.5) is the true velocity, v, of the moving object, that is

$$v = \frac{c(k) - c(n)}{T} \tag{E.6}$$

while the second term represents the velocity error denoted by e_v, that is

$$e_v = \frac{e_q(k) - e_q(n)}{T} \tag{E.7}$$

This error occurs as shown in Fig. E.1. If an object moves from any position in channel A to any position in channel B, its velocity is estimated as the slope of the dashed line connecting two centres of channels in Fig. E.1. This estimate represents the range of velocities lying between the slopes of the two diagonals (dotted lines) of the parallelogram in Fig. E.1.

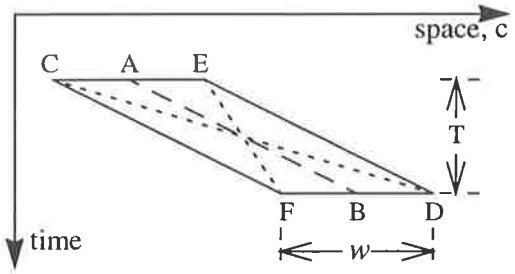


Fig. E.1 Error in estimated velocities due to the spatial quantization of detecting channels. Slopes of AB, CD, and EF represent for the estimated, possible maximum, and minimum velocities, respectively, of an object moving channel A to B.

Hence, the corresponding relative error e_r is

$$e_r = \frac{e_q(k) - e_q(n)}{c(k) - c(n)} \tag{E.8}$$

This relative error e_r has a uniform distribution with the average value of zero and the maximum value of w/D, where D is the true travelled distance, that is

$$D = c(k) - c(n) \tag{E.9}$$

Thus, the variance of the relative error is

$$\sigma^2 = E\{e_r^2\} = \frac{D}{2w} \int_{-\frac{w}{D}}^{\frac{w}{D}} e_r^2 de_r = \frac{w^2}{3D^2} \tag{E.10}$$

and its corresponding standard deviation is

$$\sigma = \sqrt{\sigma^2} = \frac{w}{D\sqrt{3}} \tag{E.11}$$

Thus, the standard deviation of this error is inversely proportional to the distance between the two locations. So, given a resolution of the spatial quantization the distance D should be increased in order to minimise the velocity relative error. This means that the estimated velocity should be averaged over a longer period of time for a more accurate estimate as the taken time is proportional to the travelled distance for a given velocity. In other words, more template data should be accumulated along the temporal axis (Nguyen et al. 1993b). Although using a lager number of images improves the resolution of the measured distance, subsequently the accuracy of estimated velocities, the greater the amount of data used, the longer the processing time is.

Alternatively, the resolution of sampling channels might be increased to improve the accuracy of the estimates. However, the sampling resolution is limited by technologies which are used to design the sensors (see Appendix A).

Bibliography

[1] Abbott, A., Moini, A., Yakovleff, A., Nguyen, X. T., Blanksby, B., Kim, G., Bouzerdoum, A., Bogner, R.E., and Eshraghian, K. (1994), "A New VLSI Smart Sensor for Collision Avoidance inspired by Insect Vision," *Proc. SPIE Intelligent Vehicle Highway Systems,* **2344**, 105-115, Boston, Massachusetts, 2-4 November.

[2] Abbott, A., Cui, S., Eshraghian, K., McCabe, E. (1991), "Photovoltaic Gate Biasing Edge Effect in GaAs MESFETs," *Electronics Letters*, **27** (21), 1900-1902.

[3] Adelson, E. H. and J. R. Bergen (1986), "The Extraction of Spatio-temporal Energy in Human and Machine Vision," *Proc. Motion: Representation and Analysis,* Charleston, IEEE Comp. Soc. Press, 151-155.

[4] Adelson, E. H. and J. R. Bergen (1985), "Spatiotemporal Energy Models for the Perception of Motion," *J. Opt. Soc. Am. A*, **2**, 284-299.

[5] Adelson, E. H. and J. R. Bergen (1983), "Spatio-Temporal Energy Models for the Perception of Motion," *J. Opt. Soc. Am.,* **73**, 1861.

[6] Aggarwal, J. K. and Martin, W. N. (1994), Reply "The Role of R & R* in Vision: Is It a Matter of Definition?" *CVGIP: Image Understanding*, **60** (1), 100-102.

[7] Aggarwal, J. K. and Nandhakumar, N. (1988), "On the Computation of Motion from Sequences of Images - A Review," *Technical Reports*, **TR-88-2-47**, Computer and Vision Research Centre, The University of Texas at Austin, USA

[8] Ali, A. T. and Dagless, E. L. (1992), "Alternative Practical Methods for Moving Objects Detection," *Proc. IEE Int. Conf. Image Process. Applic.*, 77-80, Netherlands, 7-9 April.

[9] Aloimonos, Y. (1994), Reply "What I Have Learned," *CVGIP: Image Understanding,* **60** (1), 74-85.

[10] Aloimonos, Y. and Rosenfeld, A. (1992), "Visual Recovery," *Encyclopedia of Artificial Intelligence,* S. C. Shapiro (Ed.), 1644-1687, New York: Wiley.

[11] Aloimonos, Y. and Rosenfeld, A. (1991a), Reply "A Response to 'Ignorance, Myopia, and Naiveté in Computer Vision Systems' by Jain, R. C. and Binford, T. O.," *CVGIP: Image Understanding*, **53** (1), 120-124.

[12] Aloimonos, Y. and Rosenfeld, A. (1991b), "Computer Vision," *Science,* **253**, 1249-1254.

[13] Aloimonos, Y. (1990), "Purposive and Qualitative Active Vision," *Proc. 10th Int. Conf. Pattern Recognition,* **1**, 346-360, Atlantic, NJ, USA.

[14] Aloimonos, Y. and Shulman, D. (1989), "Integration of Visual Modules: An Extension of the Marr Paradigm," San Diego: Academic Press, Inc.

[15] Anandan, P. (1989), "Computational Framework and an Algorithm for the Measurement of Visual Motion," *Int. J. Computer Vision,* **2**, 283-310.

[16] Anandan, P. (1989), "Motion Detection and Analysis - State of The Art and Some Requirements from Robotics." *IFAC Proc.,* **10**, 347-352, Pergamon Press Inc.

[17] Ancona, N. and Poggio, T. (1993) "Optical Flow from 1D Correlation: Application to a Simple Time-To-Crash Detector," *Proc. 4th Int. Conf. Computer Vision*, Berlin, Germany, 209-214.

[18] Andreou, A. G. (1991), "Electronic Arts Imitate Life," *Nature (London)*, 354, 501.

[19] Anstis, S. M. (1980), "The Perception of Apparent Movement," *Phi. Trans. R. Soc. Lond. B*, **290**, 153-168.

[20] Arbib, M. A. and Liaw, J.-S. (1995), "Sensorimotor Transformations in the Worlds of Frogs and Robots," *Artificial Intell.,* **72**, 53-79.

[21] Åström, A. (1993), "Smart Image Sensors," *PhD Dissertations*, 319, Dept. Electrical Eng., Linköping Uni., Linköping, Sweden.

[22] Bajcsy, R. (1987), "Active Perception," *Proc. IEEE,* **76** (8), 996-1005.

[23] Baker, H. H. and Bolles, R. C. (1989), "Generalizing Epipolar-plane Image Analysis on the Spatiotemporal Surface," *Int. J. Computer Vision,* **3**, 33-49.

[24] Ballard, D. H. (1991), "Animate Vision," *Artificial Intelligence,* **48** (1), 57-86.

[25] Barlow, H. B. (1979), "Reconstructing the Visual Image in Space and Time," *Nature (London),* **279**, 189-190.

[26] Barlow, H. B. and Levick, W. R. (1965), "The Mechanism of Directionally Selective Units in the Rabbit's Retina," *J. Physiol.,* **178**, 477-504.

[27] Barron, J. L., Fleet, D. J., and Beauchemin, S. S. (1994), "Systems and Experiments - Performance of Optical Flow Techniques," *Int. J. Computer Vision,* **12** (1), 43-77.

[28] Barron, J. L., Jepson, A. D., and Tsotsos, J. K. (1990), "The Feasibility of Motion and Structure from Noisy Time-Varying Image Velocity Information," *Int. J. Computer Vision,* **5**, 239-269.

[29] Baussenwein, B., Dittrich, A. P. M., and Fischbach, K.-F. (1992), "The Optic Lobe of *Drosophila Melanogaster*: II. Sorting of Retinotopic Pathways in the Medulla," *Cell T. Res.,* **267**, 17-28.

[30] Beare, R., Blanksby, A., and Bouzerdoum, A. (1995), "Low Level Visual Motion Processing using Local Motion Detectors", *Proc. IEEE Int. Conf. Neural Networks*, **1**, 1-6, Perth, Australia, 27 November-1 December.

[31] Bernard, T. M., Zavidovique, B. Y., and Devos, F. J. (1993), "A Programmable Artificial Retina," *IEEE J. Solid-State Circuits*, **28** (7), 789-798.

[32] Bertero, M., Poggio, T. A., and Torre, V. (1988), "Ill-Posed Problems in Early Vision," *Proc. IEEE*, **76** (8), 869-889.

[33] Bolles, R. C., Baker, H. H., and Marimont, D. H. (1987), "Epipolar-plane Image Analysis: An Approach to Determining Structure from Motion," *Int. J. Computer Vision,* **1**, 7-55.

[34] Borst, A. and Egelhaaf, M. (1989), "Principles of Visual Motion Detection," *Trend. Neurosci.*, **TINS 12** (8), 297-306.

[35] Bouzerdoum, A., Kouzani, A. Z., and Liebelt, M. J. (1995), "Motion Estimation Using a Fuzzy Neural Network," *Proc. Int. EEE Conf. Syst. Man Cybern.,* **5**, 4632-4637, Vancouver, British Columbia, Canada, 22-25 October.

[36] Bouzerdoum, A., Moini, A., Yakovleff, A., Nguyen, X. T., Bogner, R.E., and Eshraghian, K. (1994), "A Smart Visual Micro-Sensor," *Proc. Int. EEE Conf. Syst. Man Cybern.,* **1**, 276-279, San Antonio, Texas, USA, 2-5 October.

[37] Bouzerdoum, A. (1993), "The Elementary Movement Detection Mechanism in Insect Vision," *Phi. Trans. R. Soc. Lond. B,* **339**, 375-384.

[38] Bouzerdoum, A. (1991), "Nonlinear Lateral Inhibitory Neural Networks: Analysis and Application to Motion Detection," PhD Dissertation, University of Washington, Seatle, WA.

[39] Bouzerdoum, A. and Pinter, R. B. (1993), "A Neural Network Model for Motion Detection in the Fly Visual System," *Proc. World Congress on Neural Networks*, **1**, 58-61, Portland, USA, 11-15 July.

[40] Bouzerdoum, A. and Pinter, R. B. (1992), "Nonlinear Lateral Inhibition Applied to Motion Detection in the Fly Visual System," *NonLinear Vision: Determination of Neural, Receptive Fields, Function, and Networks,* R. B. Pinter and B. Nabet (Eds.), 423-450, CRC Press, Inc.

[41] Bouzerdoum, A. and Pinter, R. B. (1991), "Modelling the Processing and Perception of Visual Motion," *Sensory Neural Networks: Lateral Inhibition,* R. B. Pinter and B. Nabet (Eds.), 47-68, CRC Press, Inc.

[42] Bouzerdoum, A. and Pinter, R. B. (1989), "Image Motion Processing in Biological and Computer Vision Systems," *Proc. SPIE Visual Communication and Image Processing IV,* **1199**, 1129-1240.

[43] Brooks, R. A. (1991), "Intelligence Without Reason," *Proc. 12th Int. Joint Conf. Artificial Intelligence,* 569-595.

[44] Brooks, R. (1989), "A Robot that Walks; Emergent Behaviours from a Carefully Evolved Network," *Neural Computation,* **1**, 253-262.

[45] Brooks R. (1986), "A Robust Layered Control System for a Mobile Robot," *IEEE J. Robot. Autom.*, **RA2** (1), 14-23.

[46] Brown, C. M. (1984), "Computer Vision and Natural Constraints," *Science*, **224**, 1299-1305.

[47] Brown, C. M. (1994), Reply "Toward General Vision," *CVGIP: Image Understanding*, **60** (1), 89-91.

[48] Buchner, E. (1984), "Behavioural Analysis of Spatial Vision in Insects," *Photoreception and Vision in Invertebrates*, M. A. Ali (Ed.), 561-621, New York - London: Plenum Press.

[49] Buchner E. (1976), "Elementary Movement Detectors in an Insect Visual System", *Biol. Cybern.,* **24**, 85-101.

[50] Burt, P. J. (1987), "Smart Sensing within a Pyramid Vision Machine," *Proc. IEEE*, **76** (8), 1006-1015.

[51] Buxton, B. F. and Buxton, H. (1984), "Computation of Optical Flow from the Motion of Edge Features in Image Sequences," *Image and Vision Computing*, **2**, 59-75.

[52] Buxton, B. F. and Buxton, H. (1983), "Monocular Depth Perception from Optical Flow by Space Time Signal Processing," *Proc. R. Soc. Lond. B*, **218**, 27-47.

[53] Chan, M., Metaxas, D., and Dickinson, S. (1994), "A New Approach to Tracking 3D Objects in 2D Image Sequences," *Proc. 12th Nat. Conf. Artificial Intelligence*, 960-965, Seatle, Washington, 31 July - 4 August.

[54] Chellappa, R. and Kashyap, R. L. (1992), "Image Understanding," *Encyclopedia of Artificial Intelligence,* S. C. Shapiro (Ed.), 641-663, New York: Wiley.

[55] Christensen, H. I. and Madsen, C. B. (1994), Reply "Purposive Reconstruction: A Reply to 'A Computational and Evolutionary Perspective on the Role of Representation in Vision' by M. J. Tarr and M. J. Black," *CVGIP: Image Understanding*, **60** (1), 103-108.

[56] Clocksin, W. F. (1980), "Perception of Surface Slant and Edge Labels from Optical Flow: A Computational Approach," *Perception,* **9**, 253-269.

[57] Cutting, J. E. (1986), "Perception with an Eye for Motion," Cambridge, MA: MIT Press.

[58] Darling, R. B., Nabet, B., Samaras, J. E., Ray, S., and Carter, E. (1988), "Epitaxial n+ GaAs Mesa-Finger Interdigital Surface Photodetectors," *IEEE Electron Device Letters*, **10** (10), 461-463.

[59] Delbrück, T. and Mead, C. (1994), "Analog VLSI Phototransduction - by Continuous-Time, Adaptive, Logarithmic Photoreceptor Circuits," *CNS Memo 30*, California Institute of Technology, USA.

[60] Delbrück, T. and Mead, C. (1991), "Time-Derivative Adaptive Silicon Photoreceptor Array," *Proc. SPIE Infrared Sensors: Detectors, Electronics, and Signal Processing,* **1541**, 92-99.

[61] DeVoe, R. D. (1980), "Movement Sensitivities of Cells in the Fly's Medulla," *J. Comp. Physiol.*, **A-138**, 93-119.

[62] Dickmanns, E. D. and Graefe, V. (1988a), "Dynamic Monocular Machine Vision," *Machine Vision and Applications,* **1**, 223-240.

[63] Dickmanns, E. D. and Graefe, V. (1988b), "Applications of Dynamic Monocular Machine Vision," *Machine Vision and Applications*, **1**, 241-261.

[64] Diorio, C., Mahajan, S., Hasler, P., Minch, B., and Mead, C. (1995), "A High-Resolution Non-Volatile Analog Memory Cell," *Proc. Int. Symp. Circuits Systems*, Washington, USA, 29 April - 3 May.

[65] van Doorn, A. J. and Koenderink, J. J. (1983), "The Structure of the Human Motion detection System," *IEEE Trans. Systems, Man, and Cybern.*, **SMC-13**, 916-922.

[66] Duda, R. O. and Hart, P. E. (1972), "Use of the Hough Transformation To Detect Lines and Curves in Pictures," *Communication of the ACM*, **15**, 11-15.

[67] Edelman, S. (1994), Reply "Representation without Reconstruction," *CVGIP: Image Understanding*, **60** (1), 92-94.

[68] Egelhaaf, M., Borst, A., and Reichardt, W. (1989), "Computational Structure of a Biological Motion Detection System as Revealed by a Local Detector Analysis in the Fly's Nervous System," *J. Opt. Soc. Am. A,* **6** (7), 1070-1087.

[69] Egelhaaf, M. and Borst, A. (1989), "Transient and Steady-State Response Properties of Movement Detectors," *J. Opt. Soc. Am. A,* **6** (1), 116-127.

[70] Elmasry, M. I. (1985), "Digital VLSI Systems: A Tutorial," *Digital VLSI Systems*, M. I. Elmasry (Ed.), 3-9, New York: IEEE Press.

[71] Fischler, M. A. (1994), Reply "The Modelling and Representation of Visual Information," *CVGIP: Image Understanding*, **60** (1), 98-99.

[72] Fleet, D. J. and Jepson, A.D. (1990), "Computation of Component Image Velocity from Local Phase Information," *Int. J. Computer Vision,* **5**, 77-104.

[73] Franceschini, N., Pichon, J. M., and Blanes, C. (1992), "From Insect Vision to Robot Vision," *Phi. Trans. R. Soc. Lond. B,* **337**, 283-294.

[74] Franceschini, N., Riehle, A., and Le Nestour, A. (1989), "Directionally Selective Motion Detection by Insect Neurons," *Facets of Vision,* D.G. Stavenga and R. C. Hardie (Eds.), Berlin: Springer Verlag, 360-390.

[75] Foster, D. H. (1971), "A Model of the Human Visual System in Its Response to a Certain Classes of Moving Stimuli," *Biol. Cybern.,* **8**, 69-84.

[76] Ghezzi, C., Jazayeri, M., and Mandrioli, D. (1991), "Fundamentals of Software Engineering," New Jersey: Prentice-Hall, Inc.

[77] Gibson, J. J. (1950), "The Perception of the Visual World," Boston: Houghton-Mifflin.

[78] Hanahara, K., Maruyama, T., and Uchiyama, T. (1988), "A Real-Time Processor for the Hough Transform," *IEEE Trans. Pattern Anal. Mach. Intell.*, **10**, 121-125.

[79] Hausen, K. and Egelhaaf, M. (1989), "Neural Mechanism of Visual Course Control in Insects," *Facets of Vision,* D.G. Stavenga and R. C. Hardie (Eds.), Berlin: Springer Verlag, 391-424.

[80] Hausen, K. (1984), "The Lobular Complex of the Fly: Structure, Function, and Significance in Visual Behaviour," *Photoreception and Vision in Invertebrates*, M. A. Ali (Ed.), New York - London: Plenum Press, 523-559.

[81] Hausen, K. (1982), "Motion Sensitive Interneurons in the Optomotor System of the Fly - II. The Horizontal Cells: Receptive Field Organization and Response Characteristics," *Biol. Cybern.*, **46**, 67-79.

[82] Hausen, K. (1981), "Monocular and Binocular Computation of Motion in the Lobula Plate of the Fly," *Dtsch. Zool. Ges.,* **74**, 49-70.

[83] Hecht, E. (1990), "Optics," Reading, Second Edition.

[84] Heeger, D. J. (1988), "Optical Flow from Spatiotemporal Filters," *Int. J. Computer Vision,* **2**, 279-302.

[85] Heeger, D. J. (1987a), "Model for the Extraction of Image Flow," *J. Opt. Soc. Am. A*, **4**, 1455-1471.

[86] Heeger, D. J. (1987b), "Optical Flow from Spatiotemporal Filters," *Proc. 1st Int. Conf. Computer Vision,* London, 181-190.

[87] Hildreth, E. C. and Koch, C. (1987), "The Analysis of Visual Motion: from Computational Theory to Neuronal Mechanisms," *Annu. Rev. Neurosci.,* **10**, 477-533.

[88] Hildreth, E. C. (1984a), "The Computation of the Velocity Field," *Proc. R. Soc. Lond. B*, **221**, 189-220.

[89] Hildreth, E. C. (1984b), "Computations Underlying the Measurements of Visual Motion," Artificial Intelligence, **23**, 309-354.

[90] Horn, B. K. P. and Schunck, B. G. (1981), "Determining Optical Flow," *Artificial Intell.*, **17**, 185-203.

[91] Horridge, G. A. (1992) "What can Engineers Learn from Insect Vision?," *Phi. Trans. R. Soc. Lond. B,* **337**, 271-282.

[92] Horridge, G. A. (1991a), "The Compromise between Seeing Spatial Layout and Making Visual Discriminations," *Current Science,* **60**, 686-693.

[93] Horridge, G. A. (1991b), "Ratios of Template Responses as The Basic of Semivision," *Phi. Trans. R. Soc. Lond. B,* **331**, 189-197.

[94] Horridge, G. A. and Sobey, P. (1991), "An Artificial Seeing System Copying Insect Vision," *Int. J. Optoelectronics,* **6**, 177-193.

[95] Horridge, G. A. (1990), "A Template Theory to Relate Visual Processing to Digital Circuitry." *Proc. R. Soc. Lond. B,* **239**, 17-33.

[96] Horridge, G. A. (1987), "The Evolution of Visual Processing and the Construction of Seeing Systems," *Proc. R. Soc. Lond. B,* **230**, 279-292.

[97] Horridge, G. A. (1986), "A Theory of Insect Vision: Velocity Parallax," *Proc. R. Soc. Lond. B,* **229**, 13-27.

[98] Huang, C.-L. and Chen, Y.-T. (1995), "Motion Estimation Method Using a 3D Steerable Filter," *Image and Vision Computing*, **13**, 21-32.

[99] Huang, L. and Aloimonos, Y. (1994), "How Normal Flow Constraints Relative Depth for an Active Observer," *Image and Vision Computing*, **12** (7), 435-445.

[100] Huang, T. S. and Netravali, A. N. (1994), "Motion and Structure from Feature Correspondences: A Review," *Proc. IEEE,* **82**(2), 252-268.

[101] Iga, K. (1980), "Theory for Gradient-Index Imaging," *Applied Optics*, **19** (7), 1039-1043.

[102] Illingworths, J. and Kittler, J. (1988), "A Survey of the Hough Transform," *Computer Vision Graphics Image Process.*, **44**, 87-116.

[103] Jähne, B. (1993), "Spatio-Temporal Image Processing: Theory and Scientific Applications," Berlin - New York: Springer-Verlag.

[104] Jain, A. K. (1989), "Fundamentals of Digital Image Processing," Englewood Cliffs, New Jersey: Prentice Hall.

[105] Jain, R. (1994), Reply "Expansive Vision," *CVGIP: Image Understanding*, **60** (1), 86-88.

[106] Järvilehto, M. (1985), "The Eye: Vision and Perception," *Comprehensive Insect Physiology Biochemistry and Pharmacology - Nervous System: Sensory*, **6**, 355-429, G. A. Kerkut and L. I. Gilbert (Eds.), Oxford - New York: Pergamon Press Ltd.

[107] Jolion, (1994) "Computer Vision Methodologies," *CVGIP: Image Understanding*, **59**, 53-71.

[108] Kouzani, A. Z., Bouzerdoum, A., and Liebelt, M. J. (1995), "Fuzzy Motion Estimation," *Aust. J. Intell. Inform. Process. Systems.*, **2** (3), 46-55.

[109] Kouzani, A. Z. (1995), "A Fuzzy Neural Network and its Application to Motion Detection and Velocity Estimation," *M. Eng. Sc. Thesis*, Dept. EEE, University of Adelaide, South Australia.

[110] Kilger, M. (1992), "Video-Based Traffic Monitoring," *Proc. IEE Int. Conf. Image Process. Applic.*, 89-92, Netherlands, 7-9 April.

[111] Kirschfield, K. (1972), "The Visual System of *Musca*: Studies of Optics, Structure and Fusion," *Information Processing in the Visual Systems of Anthropods*, R. Wehner (Ed.), Berlin: Springer-Verlag, 61-74.

[112] Kobayashi, H., Matsumoto, T., Yagi, T., and Tanaka, K. (1995), "Light-Adaptive Architectures for Regularization Vision Chips," *Neural Networks*, **8** (1), 87-101.

[113] Koch, C. (1989), "Seeing Chips: Analog VLSI Circuits for Computer Vision," *Neural Computation*, **1**, 184-200.

[114] Kunze, P. and Hausen, K. (1971), "Inhomogeneous Refractive Index in the Crystalline Cone of a Moth Eye," *Nature (London)*, **231**, 392-393.

[115] Laughlin, S. B. (1989), "The Role of Sensory Adaptation in the Retina," *J. Exp. Biol.*, **146**, 39-62.

[116] Laughlin, S. B. (1989), "Coding Efficiency and Design in Visual Processing," *Facets of Vision*, D. G. Stavenga and R. C. Hardie (Eds.), Berlin: Springer-Verlag, 186-212.

[117] Laughlin, S. B. (1987), "Form and Function in Retinal Processing," *Trends Neurosci.*, **TINS 10** (11), 478-483.

[118] Laughlin, S. B., Howard, J., and Blakeslee, B. (1987), "Synaptic Limitations to Contrast Coding in the Retina of Blowfly Calliphora," *Proc. R. Soc. Lond. B*, **231**, 437-467.

[119] Laughlin, S. B. (1984), "The Roles of Parallel Channels in Early Visual Processing by the Arthropod Compound Eye," *Photoreception and Vision in Invertebrates*, M. A. Ali (Ed.), New York - London: Plenum Press, 457-481

[120] Laughlin, S. B. and Hardie, R. C. (1978), "Common Strategies for Light Adaptation in the Peripheral Visual Systems of Fly and Dragonfly," *J. Comp. Physiol. A*, **128**, 319-340.

[121] Lehrer, M. (1994), Minireview "Spatial Vision in the Honeybee: the Use of Different Cues in Different Tasks," *Vision Res.*, **34** (18), 2363-2385.

[122] Lehrer, M. and Srinivasan, M. V. (1994), Short Communication "Active Vision in Honeybees: Task-Oriented Suppression of Innate Behaviour," *Vision Res.*, **34** (4), 511-516.

[123] Lehrer, M., Srinivasan, M. V., Zhang, S. W., and Horridge, G. A. (1988), "Motion Cues Provide the Bee's Visual World with a Third Dimension," *Nature (London),* **332,** 356-357.

[124] Levine, M. D. (1985), "Vision in Man and Machine," San Francisco: McGraw-Hill.

[125] Li, H. and Wang, J. (1993), "Computation Optical Flow with a Recurrent Neural Network," *Int. J. Pattern Recog. Artificial Intelligence,* **7**, 801-814.

[126] Limb, J. O. and Murphy, J.A. (1975), "Estimating the Velocity of Moving Images in Television Signals.," *Computer Graphics Image Process.,* **4**, 311-327.

[127] Liou, S.-P. and Jain, R. C. (1989), "Motion Detection in Spatio-Temporal Space," *Computer Vision Graphics Image Process.*, **45**, 227-250.

[128] Longuet-Higgins, H. C. and Prazdny, K. (1980), "The Interpretation of a Moving Retina Image," *Proc. R. Soc. Lond. B,* **208**, 385-397.

[129] Marchand, E. W. (1986), "Aberrations of Wood and GRIN Rod Lenses," *Applied Optics*, **25** (19), 3413-3417.

[130] Marmarelis, P. Z. and McCann, G. D. (1973), "Development and Application of White-Noise Modelling Techniques for Studies of Insect Visual Nervous System," *Kybernetics*, **12**, 74-90.

[131] Marr, D. (1982), "Vision," New York: Freeman.

[132] Marr, D. C. and Ullman, S. (1981), "Direction Selectivity and Its Use in Early Visual Processing," *Proc. R. Soc. Lond. B,* **211**, 151-180.

[133] Marr, D. and Hildreth, E. C. (1980), "Theory of Edge Detection," *Proc. R. Soc. Lond. B*, **207**, 187-217.

[134] Mahowald, M. A. and Douglas, R. (1991), "A Silicon Neuron," *Nature (London)*, **354**, 515-518.

[135] Mahowald, M. A. (1991), "Silicon Retina with Adaptive Photoreceptors," *Proc. SPIE Visual Inf. Process.: from Neurons to Chips*, **1473**, 52-58.

[136] Mahowald, M. A. and Mead, C. A (1991), "The Silicon Retina," *Scientific American*, **264** (5), 40-46.

[137] Mead, C. (1990), "Neuromorphic Electronic Systems," *Proc. IEEE,* **78** (10), 1629-1636.

[138] Mead, C. (1989a), "Analog VLSI and Neural Systems," Reading, MA: Addison-Wesley.

[139] Mead, C. (1989b), "Adaptive Retina," *Analog VLSI Implementation of Neural Systems,* C. Mead and M. Ismail (Eds.), Boston - London: Kluwer Academic Pub., 239-246.

[140] Mead, C. A. and Mahowald, M. A. (1988), "A Silicon Model of Early Visual Processing," *Neural Networks*, **1**, 91-97.

[141] Meinertzhagen, L. A. and O'Neil, S. D. (1991), "Synaptic Organization of Columnar Elements in the Lamina of the Wild Type in *Drosophila Melanogaster,*" *J. Comp. Neurol.*, **305**, 232-236.

[142] Melles Griot, "Optics Guide5,"

[143] Michael, C. R. (1968), "Receptive Field of Single Optic Nerve Fibres in a Mammal with an All-cone Retina. II. Directionally Selective Units," *J. Neurophysiology,* **38**, 257-267.

[144] Moini, A., Bouzerdoum, A., Eshraghian, K., Yakovleff, A., and Nguyen, X. T. (1995a), "The Architecture of an Insect Vision Based VLSI Motion Detection Chip," *Proc. 13th Australian Microelectronics*, 68-73, Adelaide, 16-19 July.

[145] Moini, A., Bouzerdoum, A., Eshraghian, K., Yakovleff, A., and Nguyen, X. T., Blanksby, A., Richard, B., Bogner, R. E., and Abbott, D. (1995b), "The Architecture and Analog Building Blocks of an Insect Vision Based VLSI Motion Detection Chip," Submitted to *IEEE J. Solid-State Circuits*.

[146] Moini, A. (1994), "Design of a VLSI Motion Detector Based Upon the Insect Visual System," *M. Eng. Sc. Thesis*, Dept. EEE, University of Adelaide, South Australia.

[147] Moini, A., Bouzerdoum, A., Yakovleff, A., Abbott, D., Kim, O., Eshraghian, K., and Bogner, R. E. (1993), "An Analog Implementation of Early Visual Processing in Insects." *Proc. Int. Symposium on VLSI Technology, Systems, and Applications,* 283-287, Taipei, Taiwan, 12-14 May.

[148] Moore, A. and Koch, C. (1991), "A Multiplication Based Analog Motion Detection Chip," *Proc. SPIE Visual Inf. Process.: from Neurons to Chips,* **1473**, 66-75.

[149] Moulden, B. and Begg, H. (1986), "Some Tests of the Marr-Ullman Model of Movement Detection," *Perception*, **15**, 139-155.

[150] Mueller, G. C. and Rudolph, M. (1972), "Light and Vision," *Time-Life International* (Nederland) N. V., 53-53.

[151] Nagel, H. H. and Enkerman, W. (1986), "An Investigation of Smoothness Constraints for the Estimation of Displacement Vector Fields from Image Sequences," *IEEE Trans. Pattern Anal. Mach. Intell.,* **8**, 565-593.

[152] Nakayama, K. (1985), "Biological Image Motion Processing: A Review," *Vision Res.,* **25**, 625-660.

[153] Nelson, R. C. and Aloimonos, Y. (1989), "Obstacle Avoidance Using Flow Field Divergence," *IEEE Trans Pattern Anal. Mach. Intell.*, **11**, 1102-1106.

[154] Nesi, P. (1993), "Variational Approach to Optical Flow Estimation Managing Discontinuities," *Image and Vision Computing*, **11**, 419-439.

[155] Nguyen, X.T., Bouzerdoum, A., and Moini, A. (1996), "Velocity Measurement Using a Smart Micro-Sensor," Submitted to *Int. Sym. Robotics Cybern.*, Lille, France, 9 -12 July.

[156] Nguyen, X.T., Bouzerdoum, A., Bogner, R.E., Moini, A., and Eshraghian, K (1995), "Feature Representation of Motion Trajectories," *Proc. IEEE Int. Conf. Neural Networks*, **6**, 2922-2927, Perth, Australia, 27 November-1 December.

[157] Nguyen, X.T., Bouzerdoum, A., Yakovleff, A., Moini, A., Bogner, R.E., and Eshraghian, K. (1994a), "A VLSI Robotic Micro-Sensor: Range-Finder from Self Motion," *Proc. Int. Conf. on Machine Vision in Practice,* 78-83, Toowoomba, Australia, 13-15 September.

[158] Nguyen, X.T., Yakovleff, A., Moini, A., Eshraghian, K., Bouzerdoum, A., and Bogner, R.E. (1994b), "VLSI Architecture of a Low Computation Load Processor for a Visual System," *Proc. European Design and Test Conference,* **User Forum**, 85-89, Paris, France, 28 February - 3 March.

[159] Nguyen, X. T., Eshraghian, E., Moini, A., Bouzerdoum, A., Yakovleff, A., Abbott, D., and Bogner, R. E. (1993a), "An Implementation of a Smart Visual Micro-Sensor based upon Insect Vision," *Proc. 12th Australian Microelectronics,* 129-134, Gold Coast, Australia, 5-8 October.

[160] Nguyen, X. T., Bouzerdoum, A., Bogner, R. E., Eshraghian, K., Abbott, D., and Moini, A. (1993b), "The Stair-Step Tracking Algorithm for Velocity Estimation," *Proc. 1st Aust. New-Zeal. Conf. Intell. Infor. Systems*, 412-416, Perth, Australia, 3-5 December.

[161] Nilsson, D. E. (1989), "Optics and Evolution ofthe Compound Eye.," *Facets of Vision,* D.G. Stavenga and R. C. Hardie (Eds.), Berlin: Springer Verlag, 30-73.

[162] Nitzan, D. (1988), "Three-Dimensional Vision Structure for Robots Applications," *IEEE Trans. Pattern Anal. Mach. Intell.,* **10** (3), 291-309.

[163] Ogata, S., Ishida, J., and Sasano, T. (1994), "Optical Sensor Array in an Artificial Compound Eye," *Optical Engineering*, **33** (11), 3649-3655.

[164] Osorio, D. and Sobey, P. J. (1992), "Insect Vision as Model for Machine Vision," *Proc. SPIE Intelligent Robots and Computer Vision XI: Biological, Neural Net, and 3-D Methods*, **1826**, 253-260.

[165] Papanikolopoulos, N. P., Khosla, P. K., and Kanade, T. (1993), "Visual Tracking of a Moving Target by a Camera Mounted on a Robot: A Combination of Control and Vision," *IEEE Trans. Robotics Automation,* **9** (1), 14-35.

[166] Pichon, J. M., Blanes, C., and Franceschini, N. (1989), "Visual Guidance of a Mobile Robot Equipped with a Network of Self-Motion Sensors," *Proc. SPIE Mobile Robots IV*, **1195**, 44-53.

[167] Poggio, T., Gamble, E. B., and Little, J. J. (1988), "Parallel Integration of Vision Modules," *Science*, **242**, 436-440.

[168] Poggio, T. and Koch, C. (1985), "Ill-posed Problems in Early Vision: from Computational Theory to Analogue Networks," *Proc. R. Soc. Lond. B*, **226**, 303-323.

[169] Poggio, T., Torre, V., and Koch, C. (1985), "Computational Vision and Regularization Theory," *Nature (London)*, **317**, 314-319.

[170] Poggio, T. and Reichardt, W. (1973), "Considerations on Models of Movement Detection", *Kybernetik*, **13**, 223-227.

[171] Potter, J. L. (1977), "Scene Segmentation Using Motion Information," *Computer Graphics Image Process.*, **6**, 558-581.

[172] Prazdny, K. (1980), "Egomotion and Relative Depth Map from Optical Flow," *Biol. Cybern.*, **36**, 87-102.

[173] Ramachandran, V. S. and Anstis, S. M. (1986), "The Perception of Apparent Motion," *Scientific American*, **254**, 102-109.

[174] Reichardt, W. (1987), "Evaluation of Optical Motion Information by Movement Detectors," *J. Comp. Physiol.*, **A 161**, 533-547.

[175] Reichardt, W. (1986), "Processing of Optical Information by the Visual systems of the Fly," *Vision Res.*, **26** (1), 113-126.

[176] Reichardt, W. and Poggio, T. (1976), "Visual Control of Orientation Behaviour in the Fly," *Quart. Reviews Biophysics*, **9** (3), 311-173.

[177] Reichardt, W. (1961), "Autocorrelation, a Principle for the Evaluation of Sensory Information by the Nervous System," *Sensory Communication*, W.A. Rosenblith (Ed.), Cambridge, MA: MIT Press, 303-317.

[178] Riehle, A. and Franceschini, N. (1984), "Motion Detection in Flies: Parametric Control over ON-OFF Pathways," *Expl. Brain Res.*, **54**, 390-394.

[179] Risse, T. (1989), "Hough Transform for Line Recognition: Complexity of Evidence Accumulation and Cluster Detection," *Computer Vision Graphics Image Process.*, **46**, 327-345.

[180] Sakamoto, T. (1984), "Model for Spherical Aberration in a Single Radial Gradient-Rod Lens," *Applied Optics*, **23** (11), 1707-1710.

[181] Sandini, G. and Grosso, E. (1994), Reply "Why Purposive Vision?" *CVGIP: Image Understanding*, **60** (1), 109-112.

[182] van Santen, J. P. H. and Sperling, G. (1985), "Elaborated Reichardt Detectors," *J. Opt. Soc. Am. A*, **2**, 300-321.

[183] van Santen, J. P. H. and Sperling, G. (1984), "Temporal Covariance Model of Human Motion Detection," *J. Opt. Soc. Am. A*, **1** (5), 451-473.

[184] Schneider, C. and Card, H. (1991), "Analog CMOS Synaptic Learning Circuits Adapted from Invertebrate Biology," *IEEE Trans. Circuits Systems*, **38** (12), 1430-1438.

[185] Sejnowski, T. J. and Churchland, P. S. (1992), "Silicon Brains," BYTE, 17 (10), 137-146.

[186] Shaw, S. R. (1989), "The retina-lamina pathway insects, particularly *Diptera*, viewed from an evolutionary perspective," *Facets of Vision*, D. G. Stavenga & R. C. Hardie (Eds.), Springer-Verlag, 186-212.

[187] Shin, K. G. and Ramanathan, P. (1994), "Real-time Computing: A New Discipline of Computer Science and Engineering," *Proc. IEEE,* **82** (1), 6-24.

[188] Singh, A. (1992), "Optic Flow Computation: A Unified Perspective," IEEE Computer Society Press.

[189] Singh, A. (1990), "An Estimation-Theoretic Framework for Image Flow Computation," *Proc. 3rd Int. Conf. Computer Vision,* 168-177, Osaka, Japan.

[190] Sivilotti, M. A., Mahowald, M. A., and Mead, C. A. (1987), "Real-Time Visual Computations Using Analog CMOS Processing Arrays," *Proc. Stanford Conf. Advanced Res. VLSI*, P. Losleben (Ed.), Cambridge, MA: MIT Press, 295-312.

[191] Sklansky, J. (1978), "On The Hough Technique for Curve Detection," *IEEE Trans. Computer*, **C-27**, 923-936.

[192] Sobel, E. C. (1990a), "The Locust's Use of Motion Parallax to Measure Distance," *J. Comp. Physiol. A*, **167**, 579-588.

[193] Sobel, E. C. (1990b), "Depth Perception by Motion Parallax and Paradoxical Parallax in the Locust," *Naturwissenschaften*, **77**, 241-243.

[194] Sobey, P. and Horridge, G. A. (1990), "Implementation of the Template Model of Vision," *Proc. R. Soc. Lond. B*, **240**, 189-197.

[195] Sobey, P. (1990), "Determining Range Information from Self Motion - The Template Model," *Proc. SPIE Intelligent Robots and Computer Vision IX: Neural, Biological, and 3-D Methods,* **1382**, 123-131.

[196] Spetsakis, M. E. and Aloimonos, Y. (1988), "Optimal Computing of Structure From Motion Using Point Correspondencse in Two Frames," *Proc. 2nd Int. Conf. Computer Vision,* 449-453, Florida, USA.

[197] Srinivasan, M. V. and Zhang, S. W. (1993), "Evidence for Two Distinct Movement Detecting Mechanism in Insect Vision," *Naturwissenschaften*, **80**, 38-41.

[198] Srinivasan, M. V. (1992a), "How Flying Bees Compute Range from Optical Flow: Behavioral Experiments and Neural Models," *NonLinear Vision: Determination of Neural, Receptive Fields, Function, and Networks,* R. B. Pinter and B. Nabet (Eds.), 353-375, CRC Press, Inc.

[199] Srinivasan, M. V. (1992b), "How Bees Exploit Optical Flow: Behavioral Experiments and Neural Models," *Phi. Trans. R. Soc. Lond. B,* **337**, 253-259.

[200] Srinivasan, M. V., Pinter, R. B., and Osorio, D. (1990), "Matched Filtering in the Visual System of the Fly," *Proc. R. Soc. Lond. B,* **240**, 279-293.

[201] Strausfeld, N. J. (1989), "Beneath the Compound Eye: Neuroanatomical Analysis and Physiological Correlates in the Study of Insect Vision," *Facets of Vision*, D. G. Stavenga & R. C. Hardie (Eds.), Berlin: Springer-Verlag, 317-359.

[202] Strausfeld, N. J. (1984), "Functional Neuroanatomy of the Blowfly's Visual System," *Photoreception and Vision in Invertebrates*, M. A. Ali (Ed.), 483-522, New York - London: Plenum Press.

[203] Strausfeld, N. J. (1976), *"Atlas of an Insect Brain"*, Berlin: Springer-Verlag.

[204] Tanaka, K. and Shirai, Y. (1994), "Motion Extraction by Spatiotemporal Filtering of Analytic Signal Derived from Images and Moving Area Tracking," *Syst. Comp. Japan*, **24** (10), 24-35.

[205] Tanner, J. and Mead, C. A. (1988), "An Integrated Analog Optical Motion Sensor," *VLSI Signal Processing II,* S.-Y. Kung, R. E. Owen, and J. G. Nash (Eds.), IEEE Press, 59-87.

[206] Tanner, J. and Mead, C. A. (1984), "A Correlating Optical Motion Detector," *Proc. MIT Advanced Research in VLSI*, P. Penfield Jr. (Ed.), Cambridge, MA: MIT Press, 57-64.

[207] Tarr, M. J. and Black, M. J. (1994a), Dialogue "A Computational and Evolutionary Perspective on the Role of Representation in Vision," *CVGIP: Image Understanding*, **60** (1), 65-73.

[208] Tarr, M. J. and Black, M. J. (1994b), Response to Replies "Reconstruction and Purpose," *CVGIP: Image Understanding*, **60** (1), 113-118.

[209] Thorson, J. (1966), "Small-Signal Analysis of a Visual Reflex in the Locust: II. Frequency Dependence," *Kybernetik*, **3**, 53-66.

[210] Tomlinson, W. J. (1980), "Aberrations of GRIN-Rod Lenses in Multimode Optical Fiber Devices," *Applied Optics*, **19** (7), 1117-1126.

[211] Torre, V. and Poggio, T. (1978), "A Synaptic Mechanism Possibly Underlying Directional Selectivity to Motion," *Proc. R. Soc. Lond. B,* **202**, 409-416.

[212] Tsai, R. Y. and Huang, T. S. (1984), "Uniqueness and Estimation of Three Dimensional Motion Parameters of Rigid Objects with Curved Surfaces," *IEEE Trans. Pattern Anal. Mach. Intell.,* **6**, 13-27.

[213] Tsotsos, J. K. (1994), Reply "There is No Way to Look at Vision," *CVGIP: Image Understanding*, **60** (1), 95-97.

[214] Ullman, S. (1983), "The Measurement of Visual Motion," *Trend. Neurosci.,* **TINS 6** (8), 177-179.

[215] Ullman, S. (1981), "Analysis of Visual Motion by Biological and Computer Systems." *IEEE Computer,* **14** (8), 57-69.

[216] Ullman, S. (1979a), "The Interpretation of Visual Motion," Cambridge, MA: MIT Press.

[217] Ullman, S. (1979b), "The Interpretation of Structure from Motion," *Proc. R. Soc. Lond. B,* **203**, 405-426.

[218] Vega-Riveros, J. F. and Jabbour, K. (1989) "Review of Motion Analysis Techniques," *IEE Proceedings*, **136** (6), 397-404.

[219] Verri, A., Straforini, M., and Torre, V. (1992), "Computational Aspects of Motion Perception in Natural and Artificial Vision Systems," *Proc. R. Soc. Lond. B,* **337**, 429-443.

[220] Verri, A., Girosi, F., and V Torre (1990), "Differential Techniques for Optical Flow," *J. Opt. Soc. Am. A*, **7 (5)**, 912-922.

[221] Vittoz, E. A. (1994), "Analog VLSI Signal Processing," *J. VLSI Signal Process.*, **8**, and *Analog Integrated Circuits and Signal Process.* (Jointly Published), 27-44.

[222] Vittoz, E. A. (1985), "Micropower Techniques," *Design of MOS VLSI Circuits for Telecommunications,* Y. Tsividis and P. Antognetti (Eds.), Englewood Cliffs, NJ: Prentice-Hall, pages (MORE)

[223] Warrant, E. J. and McIntyre, P. D. (1992), "The Trade-Off between Resolution and Sensitivity in Compound Eyes," *NonLinear Vision: Determination of Neural, Receptive Fields, Function, and Networks,* R. B. Pinter and B. Nabet (Eds.), 391-421, CRC Press, Inc.

[224] Watson, A. B. and Ahumada Jr., A. J. (1985), "Model of Human Visual-Motion Sensing," *J. Opt. Soc. Am. A*, **2**, 322-341.

[225] Wolf, R., Voss, A., Hein, S., and Heisenberg, M. (1992), "Can a Fly Ride a Bicycle?" *Phi. Trans. R. Soc. Lond. B,* **337**, 261-269.

[226] Wolf, R. and Heisenberg, M. (1990), "Visual Control of Straight Flight in *Drosophia Melanogaster*," *J. Comp. Physiol. A*, **167**, 269-283.

[227] Yagi, Y., Kawato, S., and Tsuji, S. (1994), "Real-Time Omnidirectional Image Sensor (COPIS) for Vision Guided Navigation," *IEEE Trans. Robotics Automation,* **10** (1), 11-22.

[228] Yakovleff, A. (1995), "Architectural Studies for Visual Processing," *PhD. Thesis*, Dept. EEE, University of Adelaide, South Australia.

[229] Yakovleff, A., Abbott, D., Nguyen, X.T., Eshraghian, K. (1995), "Obstacle Avoidance and Motion-Induced Navigation," *Proc. Workshop on Computer Architecture for Machine Perception (CAMP'95),* 384-393, Como, Italy, 18--20 September.

[230] Yakovleff, A, Moini, A.,Bouzerdoum, A., Nguyen, X.T., Bogner, R.E., and Eshraghian, K. (1993), "A Micro-Sensor based on Insect Vision." *Proc. Workshop Comp. Archi. for Machine Perception,* 137-146, USA, 15-17 December.

[231] Yakovleff, A., Nguyen, X.T., Bouzerdoum, A., Moini, A., Bogner, R.E., and Eshraghian, K. (1994), "Dual-Purpose Interpretation of Sensory Information" *Proc. IEEE Int. Conf. Robotics Automation,* 1635-1640, San Diego, USA, 8-13 May.

[232] Zaagman, W. H., Mastebroek, H. A. K. and Kuiper, J. W. (1978), "On the Correlation Model: Performance of a Movement Detecting Neural Element in the Fly Visual-System," *Biol. Cybern.*, **31**, 163-168.

[233] Zeki, S. (1992), "The Visual Image in Mind and Brain," *Scientific American*, **267** (3), 69-76.

[234] Zhang, S. W., Srinivasan, M. V., and Horridge, G. A. (1992), "Pattern Recognition in Honeybees: Local and Global Analysis," *Proc. R. Soc. Lond. B,* **248**, 55-61.

[235] Zhang, S. W. and Horridge, G. A. (1992), "Pattern Recognition in Bees: Size of Regions in Spatial Layout," *Phi. Trans. R. Soc. Lond. B,* **337**, 65-71.

[236] Zheng, J. Y. and Tsuji, S. (1992), "Panoramic Representation for Route Recognition by a Mobile Robot," *Int. J. Computer Vision*, **9** (1), 55-76.