A VLSI Sensory-Motor Architecture for an Obstacle Avoidance Task in an Unstructured Environment

C. David Claveau

A Thesis
in
The Department
of
Electrical and Computer Engineering

Presented in Partial Fulfillment of the Requirements
for the Degree of Master of Applied Science
Concordia University
Montreal, Quebec, Canada

December, 2002

Abstract

A VLSI Sensory-Motor Architecture for an Obstacle Avoidance Task in an Unstructured Environment

C. David Claveau

Obstacle avoidance is a difficult task for autonomous robots. To overcome limitations of traditional computer vision systems, some robots have made use of efficient VLSI sensory-motor systems. However, because the processing in these systems is at the pixel level, it is difficult to achieve algorithms that can deal with real-world, unstructured environments. To make these VLSI sensory-motor systems more widely applicable, new architectures and strategies are needed.

This thesis presents an architecture for a VLSI sensory-motor system designed for obstacle avoidance by a mobile robot in an unstructured environment. Drawing inspiration from biology and behavior-based robotics, the development of the architecture is guided by an emphasis on the requirements of an obstacle avoidance behavior for a mobile robot. The architecture incorporates features which enable it to deal with unstructured environments. A special foveation and weighting scheme are used to facilitate the detection of real-world objects. The sensory and motor maps of the system are aligned to relate features in the visual field to a left and a right control signal. The effectiveness of the architecture is demonstrated through computer simulation. A model of a sensory-motor system based on the architecture is used to create a realistic looking virtual environment simulator. Simulation results show that such a system is capable of efficient obstacle avoidance in an unstructured environment, while using only a small number of simple operations connected hierarchically, potentially leading to an implementation with small pixels.

Contents

List of Figures

Acronyms and Abbreviations

CCD	Charge-Coupled Device
CMOS	Complementary Metal Oxide Semiconductor
CPU	Central Processing Unit
DSP	Digital Signal Processing
EMD	Elementary Motion Detector
MOS	Metal Oxide Semiconductor
PPC	Posterior Parietal Cortex
SC	Superior Colliculus
SMPA	Sense Model Plan Act
SOC	System-On-A-Chip
VLSI	Very Large Scale Integration
WTA	Winner-Take-All

Special Symbols

$f_{nc}, f_{nd}, f_{ns}, f_{np}$	field values for center, diagonal, side and path regions at level n
w_n, h_n	width and height in pixels of the fields at level n
$I(x,y,t)$	pixel value at position x,y and time t
$c_L(t), c_R(t)$	left and right control signals
K_{nc}, K_{nd}, K_{ns}	weights for the differences between the fields for each level n
$\Delta a(t)$	change in orientation angle
K_α	scaling factor for the change in orientation angle
x_c, y_c, z_c	camera coordinate system
u, v	image coordinate system (the image plane)
u_T, v_T	size of the sensor in image coordinates
u_o, v_o	image coordinates for the projection of the smallest detectable area
σ_1	maximum dimension of the smallest field on the sensor
C	center of projection
d_{min}	distance from sensor to intersection of optical axis and ground plane
h	height of sensor (from center of projection to ground plane)
θ	vertical gaze angle
ϕ_x, ϕ_y	field of view angles
f	focal length of pinhole camera imaging model
w	width of robot body
v_L	speed of left wheel along the ground plane
v_R	speed of right wheel along the ground plane
r	turn radius for a differential steering system
b	wheel base for the differential steering system
ω	rotational velocity of the differential steering system
α	orientation angle of the robot

1

Introduction

Moving through a cluttered room without bumping into anything can be easily done by all creatures, great and small, that possess the necessary perceptual and motor capabilities. Getting a robot to perform that task has proven to be very difficult. This is partly because it is a sensory-motor task, which means that an appropriate motor action must be taken according to the signals derived from a sensory system. Sensing the environment in a timely and sufficiently detailed manner is difficult. It usually involves a large amount of

data which must then be processed quickly enough to produce a control signal for an actuator. The various creatures of the world have vision to sense their environments. Visual features can be used to distinguish the obstacles from the background, and to perceive the depth and relative position of these obstacles. Likewise, when engineering a sensory-motor system for a robot, some type of camera system is generally used to acquire images of the robot's environment. The image data are processed by a computer using various algorithms. Recently there has been growing interest in acquiring the visual signal and processing it on the same integrated circuit. With these 'smart sensors', the algorithm is embedded in their signal processing architecture and the circuits used to implement it. The work in this thesis comprises such an architecture, designed for obstacle avoidance in real-world, unstructured environments.

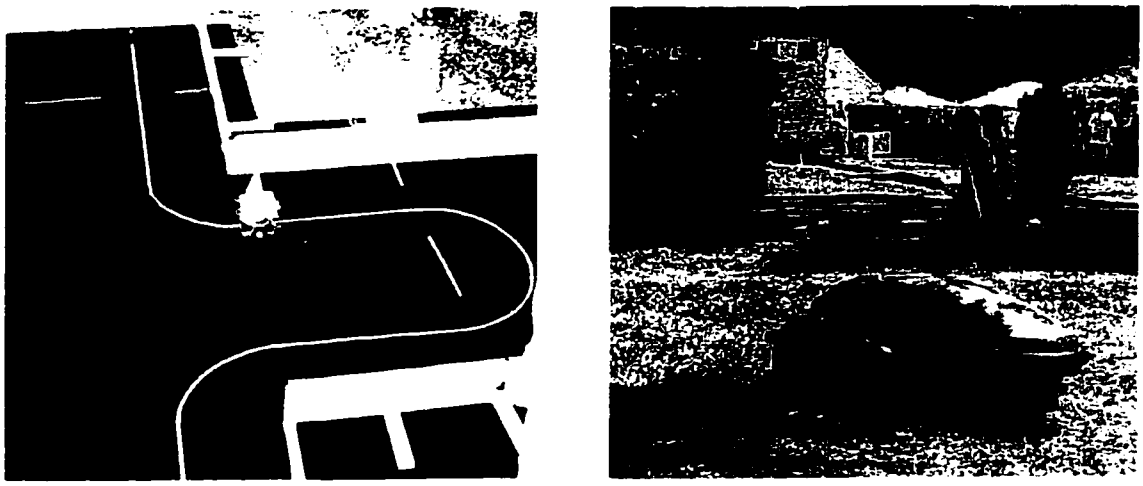
1.1 Autonomous Mobile Robots and Obstacle Avoidance

An autonomous mobile robot must operate without the aid of an external controller. The robot is self-contained, which means that it has its own sensors, its own processing units, and its own actuators to perform its tasks. Because it is mobile, it must be able to move through a certain variety of environments and avoid obstacles. Achieving that functionality within the constraints of the robot's resources is a unique and important challenge. Its importance lies in the many useful applications of these robots.

The major application areas for autonomous mobile robots are scientific, industrial, military, and consumer. Scientific applications include using mobile robots for exploration and data gathering. Industrial applications include surveillance and various tasks in factory settings. Military applications usually involve reconnaissance robots which must pass

through hostile areas. Consumer applications include service robots such as the automatic vacuum cleaning and lawn mowing robots already available on the market. All of these applications require the robot to perform reliable obstacle avoidance in its environment.

The difficult part for many of these applications is that the robot should be able to move safely and efficiently through an unstructured environment. An unstructured environment is one which has not been purposely designed or altered to facilitate the movement of a robot. Examples include natural environments like the surface of a planet, or man-made environments like a factory floor. However, if a white line is painted on the floor so that a robot can use the line to guide its movements, then it is an artificially structured environment. Fig. 1.1 shows an example of both.



(a) (b)

Fig. 1.1 Structured and unstructured environments. Shown in (a) is the top-view of a structured environment in which a robot follows a white line on a black floor. An unstructured environment is in (b). A lawn mower robot must avoid obstacles like bushes and people.

For traditional computer vision systems, performing a task such as obstacle avoidance in an unstructured environment is computationally intensive. First, the image sequence from the camera must be processed fast enough to permit the robot to move at a reasonable speed. This requires processing at very high clock rates because of the large amount of data. Secondly, extracting information like the position and size of obstacles is algorithmically complex because of the many different ways they may appear in realistic images.

Exploratory robots for scientific applications are good examples of this problem because they are often deployed in challenging and remote locations. Fig. 1.2 shows an experimental rover designed to explore the surface of the planet Mars. It uses solar cells to power its motors and its sensory equipment. Because of the great distances involved, and the great financial costs of these missions, the robot must explore the planet as efficiently and reliably as possible. Using traditional computer vision for navigation and obstacle avoidance is expensive in terms of time, power, and size. On the Sojourner rover of the Pathfinder Mars mission in 1998, the camera image sensors consumed 5% (0.75W) of the total power budget and the microprocessor-based system consumed 24% [1]. The microprocessor was mostly used to process static images while the rover remained motionless, waiting for the results [1].

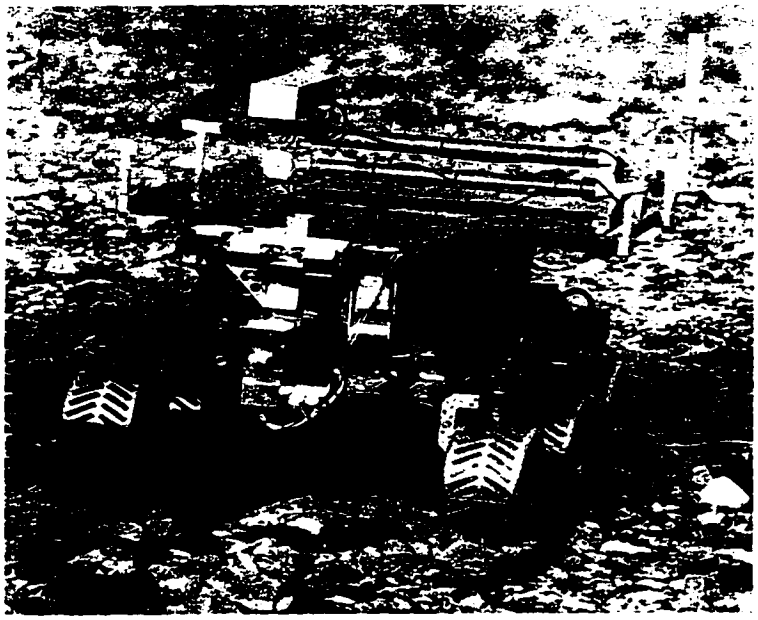


Fig. 1.2 An experimental Mars rover for planetary exploration is an example of an autonomous robot with an important and difficult task.

1.2 Obstacle Avoidance and VLSI Smart Sensors

Because of the limitations of the traditional techniques, other approaches including those based on VLSI smart sensors have been explored. A smart sensor consists of an array of pixels, each of which contains a photoreceptor and some signal processing circuitry. In the smart sensor paradigm, the per-pixel signal processing circuitry can perform various operations on the photoreceptor signals. Some types of image processing and computer vision algorithms can be implemented directly on the sensor. The algorithm is entirely determined by the architecture of the smart sensor. The architecture defines the signal processing operations to be performed and how they are interconnected across the array. For example, sometimes the processing is used to produce an image with enhanced

edges, which can then be scanned out and digitized. Sometimes the processing is used to compute a global property of the image such as the minimum/maximum brightness or an image histogram.

Often the architecture of the signal processing is derived from a study of biological systems which perform tasks related to vision, such as target tracking and motion detection. If this visual processing produces a motor control signal to follow a target or to steer a robot to avoid obstacles, then the system can be referred to as a VLSI sensory-motor system. Such a system can be used in a control loop to replace a traditional microprocessor based computer vision system which may use multiple chips for things like analog-to-digital conversion, memory and processing. In addition, high-bandwidth inter-chip connections can be eliminated by having a single chip which can take a visual signal from a scene and produce a motor control signal that can be used with minimal interfacing.

In the last 10 to 15 years, several VLSI sensory-motor systems for autonomous robots have been proposed. They are able to perform their functions efficiently because they process the visual signal in parallel. However, extracting relevant information about real-world obstacles is still a difficult task. Because of limited space, the complex algorithms used in conventional systems are not implementable using sensor arrays. Also, since the processing of a sensor array is mainly at the level of individual pixels, it is usually better suited to tasks in low-level vision such as edge detection. Thus, it is much easier to locate a white line or a spot of light on a black background than it is to locate a bush or a rock.

To overcome these difficulties, new architectures and strategies are needed to make VLSI sensory-motor systems more widely applicable to real-world robotics. The research

in this thesis is motivated by an interest in bringing together ideas from robotics, biology, and smart sensor technology to propose a new architecture for a VLSI sensory-motor system for an autonomous mobile robot. The research is based on the thinking that a confluence of ideas from these three fields can lead to a more effective solution for difficult tasks like obstacle avoidance in unstructured environments. Drawing inspiration from biological systems has become increasingly popular as an engineering strategy in the field of robotics and also in the field of smart sensors. This is particularly true for systems which must perform tasks which resemble things that humans and animals appear to perform with ease.

1.3 Objectives

The first major objective of the research in this thesis is to develop a signal processing architecture for a VLSI smart sensor intended to efficiently solve the problems of sensory-motor systems using minimal hardware resources. The architecture should lead to an effective obstacle avoidance behavior in realistic environments. It should remain effective in the presence of irregularly shaped objects with textured surfaces and many details. Also, the architecture should exhibit the qualities which make it suitable for a smart sensor implementation. One of these qualities is to use only computationally simple operations so that the systems can be easily implemented using standard VLSI technologies at a low cost.

In order to evaluate the efficiency of the architecture, modules for the simulation of the architecture need to be developed, which is the second objective of the work of this thesis. Because of the recent availability of high-performance workstations, it is possible to create detailed software simulations which permit flexible and accurate prototyping. In

this case, a virtual environment is needed to test the architecture as if it was actually mounted on an autonomous mobile robot in a realistic environment, like a room or in a factory. The simulation should model three things: the realistic environment, the imaging process, and the functionality of the architecture which determines the steering of the robot. A well constructed simulation can also be used to determine some important design parameters for an eventual implementation.

The third objective of the thesis is to provide an example implementation of the architecture using simple circuits in order to prove the feasibility of a VLSI implementation which uses standard technology.

1.4 Overview and Outline of the Thesis

This thesis presents a VLSI sensory-motor architecture which is directed at performing a useful task in an unstructured environment. The architecture is designed for an obstacle avoidance task by a mobile robot in a real-world, unstructured environment where there are no simple visual cues to follow. The architecture maps easily to the smart sensor paradigm, but incorporates special features which enable it to avoid real-world obstacles in common settings.

The features of the architecture will be developed by emphasizing the particular requirements of an obstacle avoidance task for a mobile robot. For obstacle avoidance, a control signal to steer the robot must be derived from the visual signal. To this end, a special type of foveation scheme will be developed which produces a low-pass filtering of the image to make complex objects appear more like a block of uniform intensity. Also, a bio-

logically inspired technique of aligning sensory and motor maps will be used to produce a control signal to keep the robot centered on the open pathway.

This thesis describes the development and simulation of the VLSI sensory-motor architecture in detail. In Chapter 2, relevant background material on autonomous robots and obstacle avoidance is presented, along with a brief description of some of the biological inspirations for the thesis. Some recent VLSI sensory-motor systems, related to the work in this thesis, are also presented. Chapter 3 describes the development of the architecture in detail, starting with the two basic organizing principles and ending with a complete architecture for an obstacle avoidance system. Chapter 4 describes a simple mathematical model of the architecture which is used as the basis for computer simulation. A detailed simulation scheme using 3D computer graphics is presented to evaluate the effectiveness of the architecture. Chapter 5 presents a set of circuits for the operations required by the architecture to demonstrate the feasibility of a potential implementation in a standard CMOS technology. Finally, Chapter 6 summarizes the most important points of the thesis and suggests opportunities for future research.

2

Background and Related Work

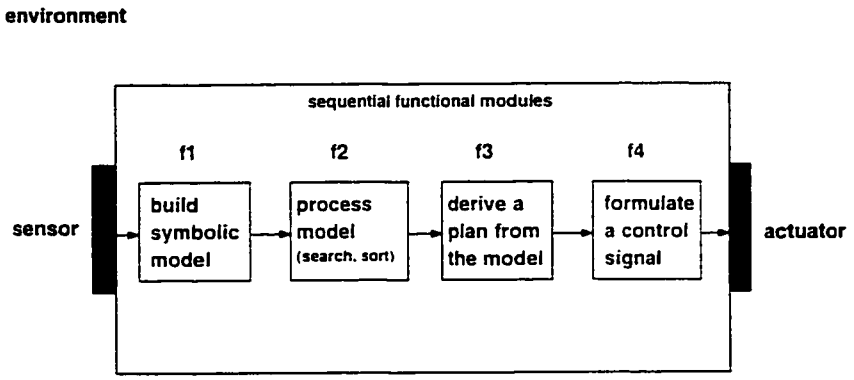
This chapter presents some background material on VLSI smart sensors and sensory-motor systems. It reviews the key historical developments in the smart sensor paradigm followed by a consideration of a set of more recent sensory-motor systems. Some of the principle influences and inspirations for the work in this thesis, such as those related to biological vision systems and behavior-based robotics, are highlighted.

2.1 Autonomous Robots and Obstacle Avoidance

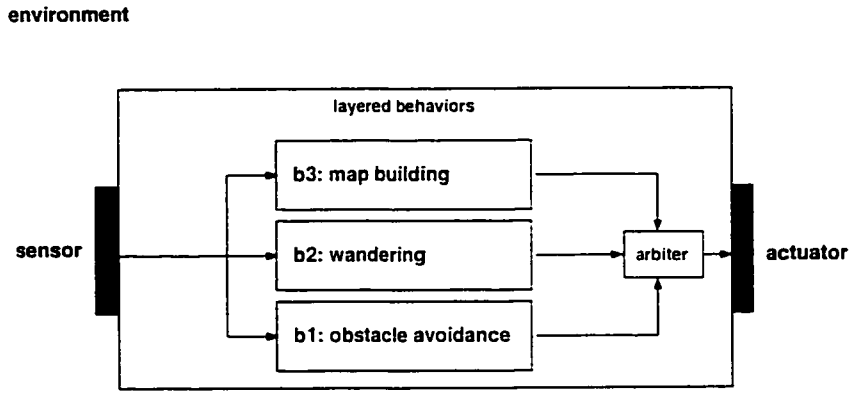
As mentioned in Chapter 1, an autonomous mobile robot must be able to move through various environments and avoid obstacles without the aid of an external controller. This type of automatic sensory-motor behavior is difficult to achieve because of the large volume of sensory data that must be processed in a timely and efficient manner. Traditional approaches have been strongly influenced by the model of computation supported by the stored-program computer. They have generally followed what is referred to as the sense-model-plan-act (SMPA) cycle. In this cycle, sensing is performed in order to construct a central, symbolic model of the robot's environment. The model, stored in computer memory, is then used as the basis for all decision making and planning for things such as obstacle avoidance or more complex behavior. It should be a detailed, comprehensive model of the environment, which is able to support the planning of actions for most of the robot's behavior. The purpose of visual sensing in these systems was to extract features from the environment which could be incorporated into the general purpose model.

There have been several problems with the SMPA approach. It is difficult for the robot to be reactive to a dynamic environment because feature extraction and model building must be executed on a large amount of incoming data and this process can be slow. Likewise for the planning stage, model-based computations like searching and sorting often require high-speed processing. Also, because planning and decision making is based on a model, the model must be kept up-to-date. Situations can arise in which the model no longer accurately reflects the state of the environment and the resulting plan could lead to disaster. As a result of these deficiencies, the traditional systems were often developed for simplified environments in which obstacles were easy to detect and locate.

A different approach, referred to as behavior-based robotics [21][22], was developed in the mid-1980s. The traditional approach decomposes the task of robot control into a sequence of functional modules which were implemented as algorithms operating on detailed data structures, as shown in Fig. 2.1 (a). The behavior-based approach, shown in Fig. 2.1 (b), decomposes the task into layers of separate 'elementary' behaviors that can interact to form the complete behavior of the robot.



(a)



(b)

Fig. 2.1 Two approaches to robot control. The traditional approach is shown in (a) and the behavior-based approach is shown in (b).

Each behavior performs a complete (though often simple) task and forms a tight coupling between perception and action. Separate sensing strategies can be tailored to each behavioral module, depending on the needs of that particular behavior. The importance of this approach to the work of this thesis lies in the following two points:

1) Sensing of the environment should be done to extract behaviorally relevant information, not to construct a central, detailed model of the world.

2) Sensing should be directly coupled to action without intervening modeling and complex data processing. Information which is directly derived from the 2D image sequence should be used for decision making and control.

Because of the way they simplify computation, these two points are applicable to the development of a sensory-motor architecture targeted to a smart sensor implementation which has limited computational resources.

An example of a behavior-based system which inspired this thesis is the obstacle avoidance system developed by Lorigo [19], based on earlier work by Horswill [20]. In their system, a camera is mounted on the front of a robot and is angled toward the ground so that the bottom of the image is assumed to show the unobstructed, 'safe' path. Using the assumption that the ground is flat and that obstacles lie on the ground, the height in the image of an obstacle will indicate its depth. The image heights of obstacles are found from left to right across the image. The robot is steered by merely computing a turn angle which is proportional to the difference between the average of the obstacle image heights on the left half of the image and the average on the right.

2.2 Imitating Biology

Drawing inspiration from biology has become a practical strategy for many fields of engineering. Biomimetic or biomorphic systems are designed to imitate biological structures and processes. Recent sensory-motor systems have showed a tendency to make use of biological models of sensory-motor coordination. Generally the models are derived from sensory-motor regions of the vertebrate brain such as the superior colliculus (SC) and the posterior parietal cortex (PPC), shown highlighted in Fig. 2.2. The figure shows the most important parts of the primate visual system. Two main pathways can be distinguished. Most of the signal from the retina follows the primary visual pathway to areas in the cerebral cortex. Part of the signal, however, follows the secondary visual pathway to the SC.

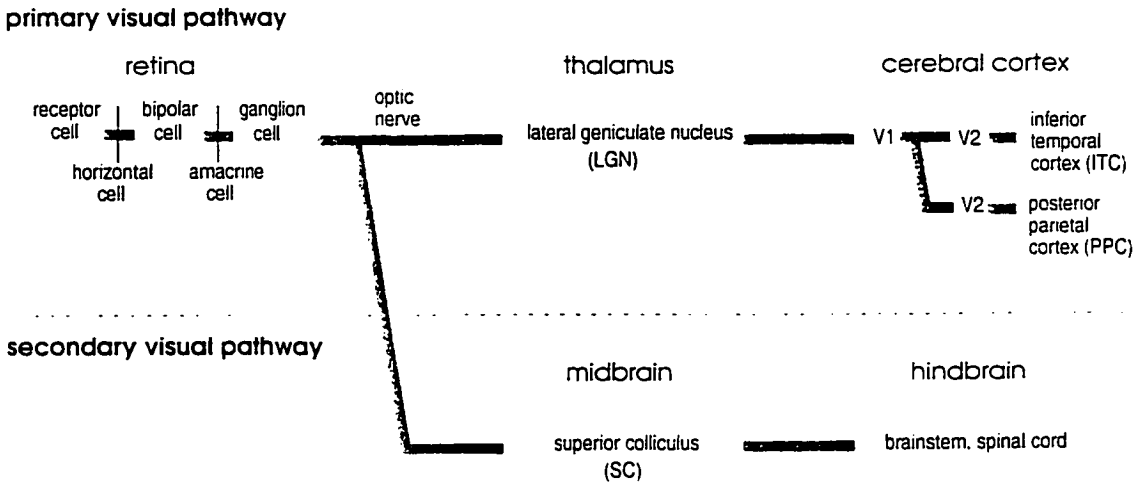


Fig. 2.2 The primary and secondary visual pathways in the brain. The superior colliculus and the posterior parietal cortex are shown highlighted.

Research has shown that the SC contains layers of neurons which receive visual input from the eyes and other senses, arranged so that they preserve the map of the visual field as it is sensed by the retina (a sensory map or a retinotopic map) [12]. The SC also contains layers of neurons which project to motor centers of the brain which control the orientation of the eyes and head. These layers are referred to as a motor map. The sensory and motor maps overlap each other and are aligned so that activity in a particular region of the sensory map will trigger activity in the motor map which will lead to an orientation toward that region of the visual field. This mechanism is thought to underlie our ability to acquire a target in the visual field and to track its movement. Similar to the superior colliculus, neurons in the PPC receive both visual signals and eye position signals. By recording the activity of these neurons it has been shown that the amplitude of their response is modulated by the position of the eye [13]. These neurons are described as having 'gain modulated' receptive fields (similar to 'weighted' signals, in signal processing terms). Based on these studies it has been postulated that gain modulation in the PPC is used to transform eye-centered coordinates to body-centered coordinates by taking into account any shift in gaze angle.

These two strategies, the alignment of sensory and motor maps and the use of weights for signals from different parts of the visual field, can be readily applied to sensory-motor systems based on the smart sensor paradigm. Chapter 3 will describe how the architecture in this thesis makes use of both strategies.

2.3 VLSI Smart Sensors

2.3.1 The Smart Sensor Paradigm

The sensory-motor architecture proposed in this thesis is based on the smart sensor paradigm. A VLSI smart sensor consists of an array of pixels, each of which contains a photoreceptor and some signal processing circuitry, as shown in Fig. 2.3. The photoreceptor array transduces the incoming pattern of light into an array of currents or voltages. The per-pixel signal processing circuitry can perform various operations on these photoreceptor signals, including some types of algorithms for image pre-processing. In the smart sensor paradigm, the algorithm is entirely determined by the signal processing architecture, which defines the operations to be performed and their interconnections across the array.

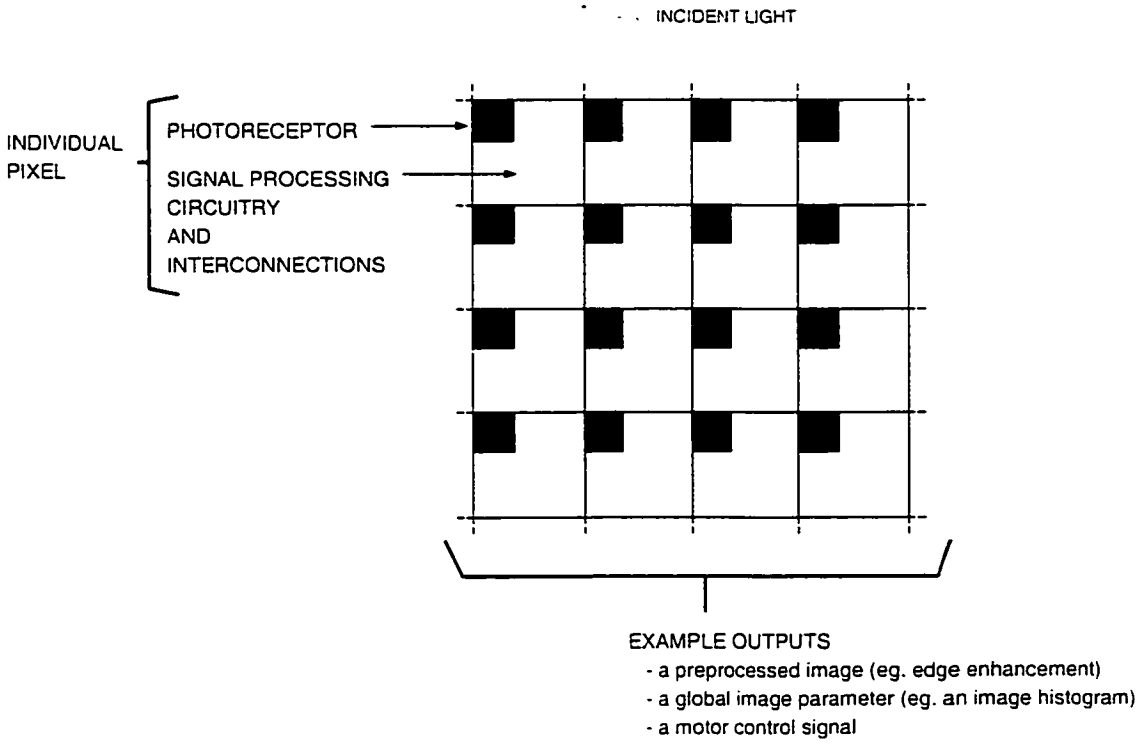


Fig. 2.3 The smart sensor paradigm. Signal processing is performed at the pixel-level, making for a highly parallel sensing and processing system on a single chip.

Some advantages of the smart sensor paradigm include an increase in processing speed due to the parallel nature of the architecture and less time spent on the transfer of image data from a sensor to a processor. Also, total system size can be reduced because most of the processing can potentially be done on a single chip. There is also a decrease in power consumption because of the reduced hardware.

To minimize the area per pixel, analog circuitry is often used in sensor arrays. A disadvantage of analog signal processing is that it results in less accuracy and a loss of precision compared with digital techniques. Also, since the processing in a smart sensor is mainly at the level of individual pixels, it is usually better suited to tasks in low-level vision. Low-level vision is mainly concerned with extracting primitive features from an image. Edge detection is an example of such an activity. Intermediate-level vision is concerned with grouping primitive features in semantically meaningful ways. Figure-ground segregation is an example.

The following subsections present a brief survey of some of the key developments in smart sensor technology, in roughly chronological order.

2.3.2 Early Solid-State Image Sensors

The idea of putting an array of phototransistors on a single chip was initially motivated by the desire to have a solid-state television camera. A 50x50 array of phototransistors was reported in 1966 by Westinghouse [2]. Other sensors based on metal-oxide semiconductor (MOS) technology were reported in the late 1960's [3][4]. Research into MOS-based sensors declined during the 1970's however, because in 1970 the CCD

(charge-coupled device) sensor was developed [5]. Its smaller pixel size, higher sensitivity, and better noise reduction made it the dominant form of integrated sensor technology.

2.3.3 Early Smart Sensors

Interest in MOS-based sensor arrays was renewed in 1981 when an optical mouse was reported by Lyon [6]. The mouse was based on a chip which is arguably the first VLSI smart sensor. It combined a sensor array with mixed analog and digital circuitry to track the movement of a fixed pattern on a mouse pad. Most importantly it showed that a standard NMOS technology could be used for both sensing photons and for analog/digital circuitry. It also showed the importance of using simple operations and conservative circuit designs when developing smart sensors.

2.3.4 Neuromorphic Sensors

In 1989, Mahowald and Mead [7][8] developed a smart sensor which was structurally and functionally similar to the outer portions of the vertebrate retina. Its architecture is shown in Fig. 2.4 along with the corresponding cell structures of the retina. In the vertebrate retina, a network of horizontal cells average the outputs of the photoreceptors spatially and temporally. The bipolar cells detect the difference between the averaged output of the horizontal cells and the input from each photoreceptor. In the silicon retina, a network of active resistors, implemented with MOS transistors, is used to emulate the averaging of the horizontal cells. Each photoreceptor circuit consists of a phototransistor with

additional differential amplifiers to compute the difference between the average computed by the resistive network and the value of the phototransistor. The result is a spatio-temporal filtering which enhances edges and their movement. In particular, the design highlights the use of averaging and differencing to bring out important features in the array. These two operations seem to be basic to both biological and electronic vision systems.

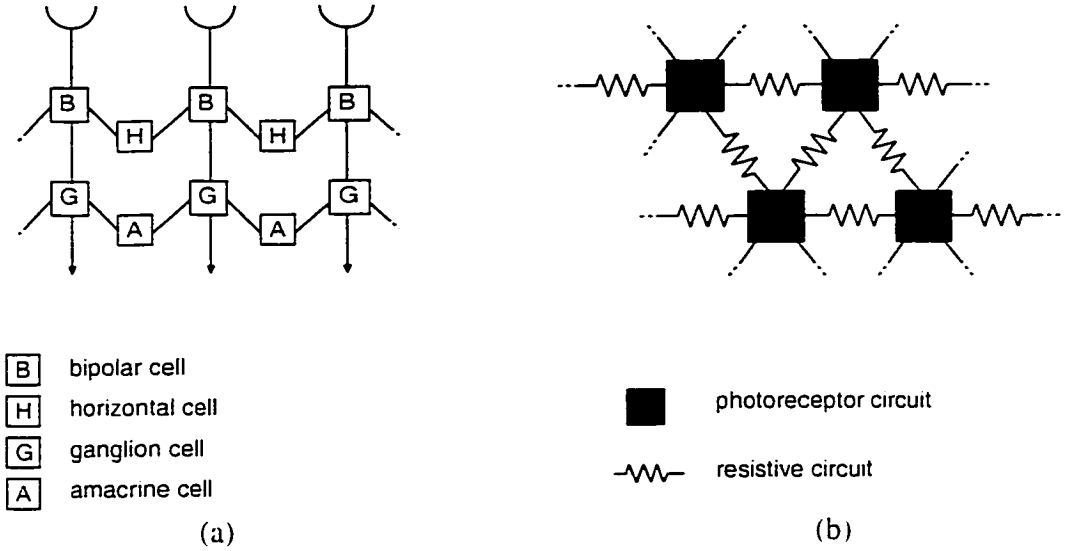


Fig. 2.4 The neuromorphic architecture of the silicon retina [8]. (a) shows the structure of the retina in which the signals from the photoreceptors are averaged by the network of horizontal cells. (b) shows the resistive network used by the silicon retina to perform a similar type of averaging.

2.3.5 Foveated Sensors

Another feature of the retina that has been used in smart sensor design is foveation. A 'foveated' sensor is one in which the density of photoreceptors decreases radially away from the center. Foveation allows for a wide field of view and while maintaining a region of high resolution near the center, without requiring too many receptors. The sensor shown in Fig. 2.5 was reported by Wodnicki [9]. It has a high resolution central region, the fovea,

and a peripheral region with decreasing resolution. In the central region, photoreceptors are uniformly spaced in a rectangle and in the periphery they are placed in a circular array. The log-polar arrangement in the periphery has been shown to simplify operations such as rotation and scaling in two dimensions [10]. Also, these types of sensors show promise for sensory-motor type systems because they can differentiate between stimuli in the peripheral region and stimuli in the central region. Unfortunately, they are difficult to design because of the nonuniform spacing and sizing of the photoreceptors, as can be seen in the figure.

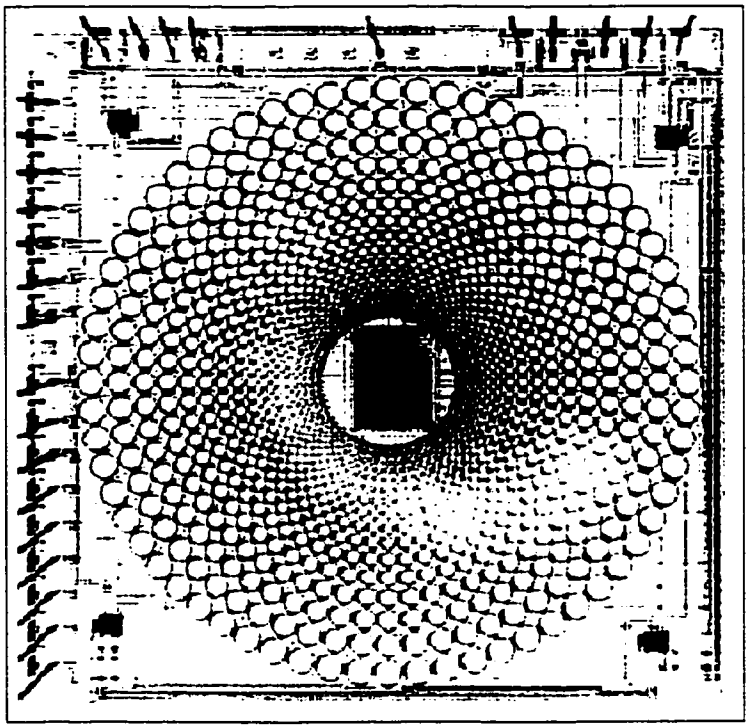
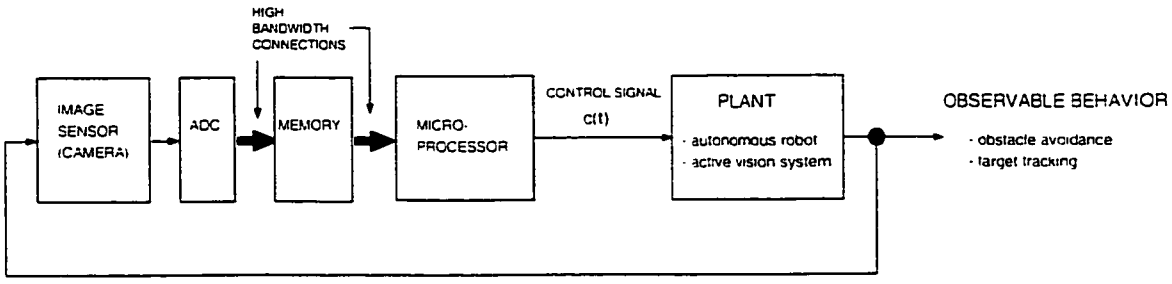


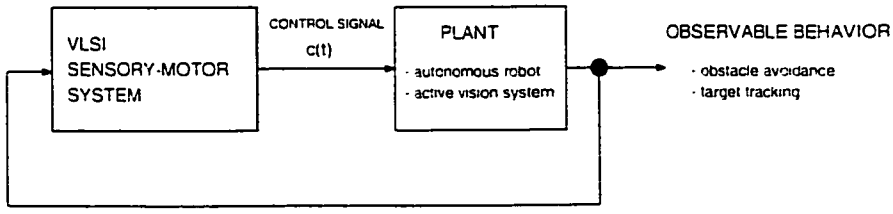
Fig. 2.5 A photomicrograph of a foveated sensor [9]. The high resolution central region is the fovea. It is surrounded by a peripheral region where pixels have different sizes and spacings depending on their radial distance from the center.

2.4 VLSI Sensory-Motor Systems

Advances in both fabrication technologies and circuit design have lead to the ability to integrate more functionality onto smart sensors. As mentioned in the introduction, a very useful type of functionality for a smart sensor is the ability to produce a motor control signal from its sensor array. The resulting VLSI sensory-motor system could be used in many applications which would benefit from visually-guided control. As shown in Fig. 2.6, compared to a traditional computer vision system consisting of a camera and a digital processor, a VLSI sensory-motor system would provide some obvious system-level advantages such as reduced size and power consumption and less movement of data through high-bandwidth connections.



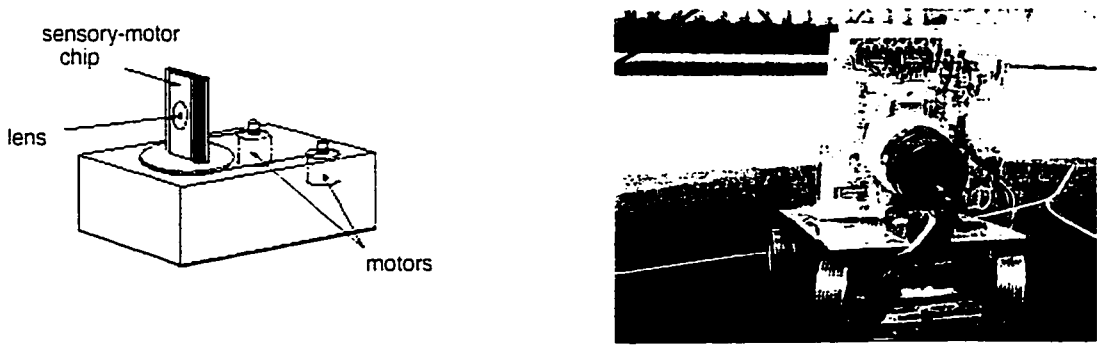
(a)



(b)

Fig. 2.6 Taking advantage of a VLSI sensory-motor system in a typical control loop. A traditional microprocessor-based computer vision system, shown in the grey box in (a), is replaced with a single-chip VLSI sensory-motor system, in (b).

VLSI sensory-motor systems are most often used to control either an active vision system or an autonomous mobile robot. In the active vision system of Fig. 2.7 (a), the sensor array is continuously reoriented by the left and right motors in order to perform tasks like target acquisition and target tracking. The sensory-motor system produces a control signal for the motors based on the two-dimensional image signal of its sensor array. In a similar way, a VLSI sensory-motor system was used in the robot in Fig. 2.7 (b) to make it follow a white line on the floor. One of the earliest examples of such a system for a robot was reported in 1991 by Clark and Friedman [11]. It performed a type of edge detection from which it created a control signal to enable an autonomous robot to perform a line-following task.



(a) (b)

Fig. 2.7 Two common test applications for VLSI sensory-motor chips. In (a) a sensory-motor chip is used to control the left and right motors of an active vision system in order to track a target. In (b) a wheeled robot uses a sensory-motor chip to follow a white line on a black floor.

2.4.1 Approaches Based on Feature Position

Recent VLSI sensory-motor systems have tended to use one of two strategies. The first approach, discussed in this section, finds the brightest or 'strongest' feature in the scene and converts its position on the sensor array to a motor control signal, usually to keep the feature centered on the array. The feature, an edge or a spot of light, is usually detected with per-pixel spatial derivative operations across the array, and the brightest feature is determined with a winner-take-all (WTA) circuit [14]. The WTA circuit suppresses all signals except the strongest, which it allows to pass.

One of the best examples of this approach is the sensory-motor robot controller of Maris and Mahowald [15]. It was used by a mobile robot in a line-following task. A similar architecture was also reported by Indiveri [16]. The essential elements of this type of architecture are shown in Fig. 2.8. The architecture consists of an array of photoreceptors followed by edge enhancing circuitry.

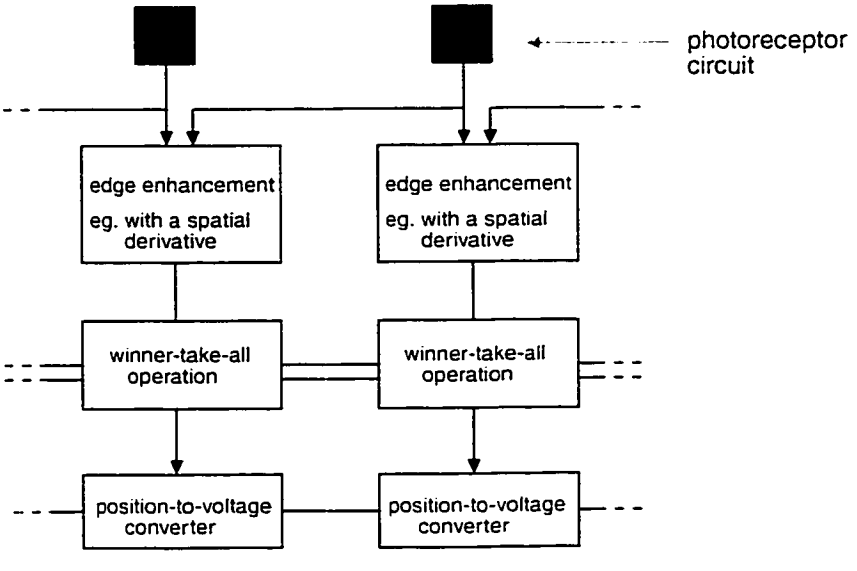


Fig. 2.8 Block diagram showing two pixels of an architecture that finds the position of an edge using a WTA operation.

The output of the edge enhancement is sent to a WTA to identify the pixel with the strongest edge. A position-to-voltage circuit then converts the position of the pixel in the array to an analog voltage which is used to keep the robot centered over a line on the floor.

2.4.2 Approaches Based on Motion Detection

The second commonly reported approach for VLSI sensory-motor systems detects any motion in the scene and then determines the direction and speed of that motion. It then produces a motor control signal to move the sensor in such a way as to match the direction and speed of the motion detected in the scene. Often, a circuit which is modeled on the elementary motion detector (EMD) thought to exist in biological vision systems, is used.

A good example of this type of architecture is the system described by Harrison [17]. Fig. 2.9 (a) shows the basic EMD used. A measure of image motion between two pixels is computed by correlating the output of one photoreceptor with the delayed output of an adjacent photoreceptor. Two adjacent photoreceptors send their outputs to temporal bandpass filters which remove constant illumination (no information) and high frequencies (noise). These signals are then delayed by using the phase lag inherent in a first-order temporal lowpass filter. Delayed channels are then correlated with nondelayed channels through a multiplication operation. The two outputs are differenced to produce a positive response for leftward motion and a negative response for rightward motion. Fig. 2.9 (b) shows how a subgroup of the array of EMDs can be spatially summed in order to reduce the effects of complex textures on the total response. Fig. 2.9 (c) shows how two side-mounted motion sensors can be compared in order to perform obstacle avoidance. As the

robot moves through its environment, nearby objects will move across a motion sensor's field of view faster than far away objects. If it steers away from the side that measures the largest velocity, it can avoid large objects and stay on the open path.

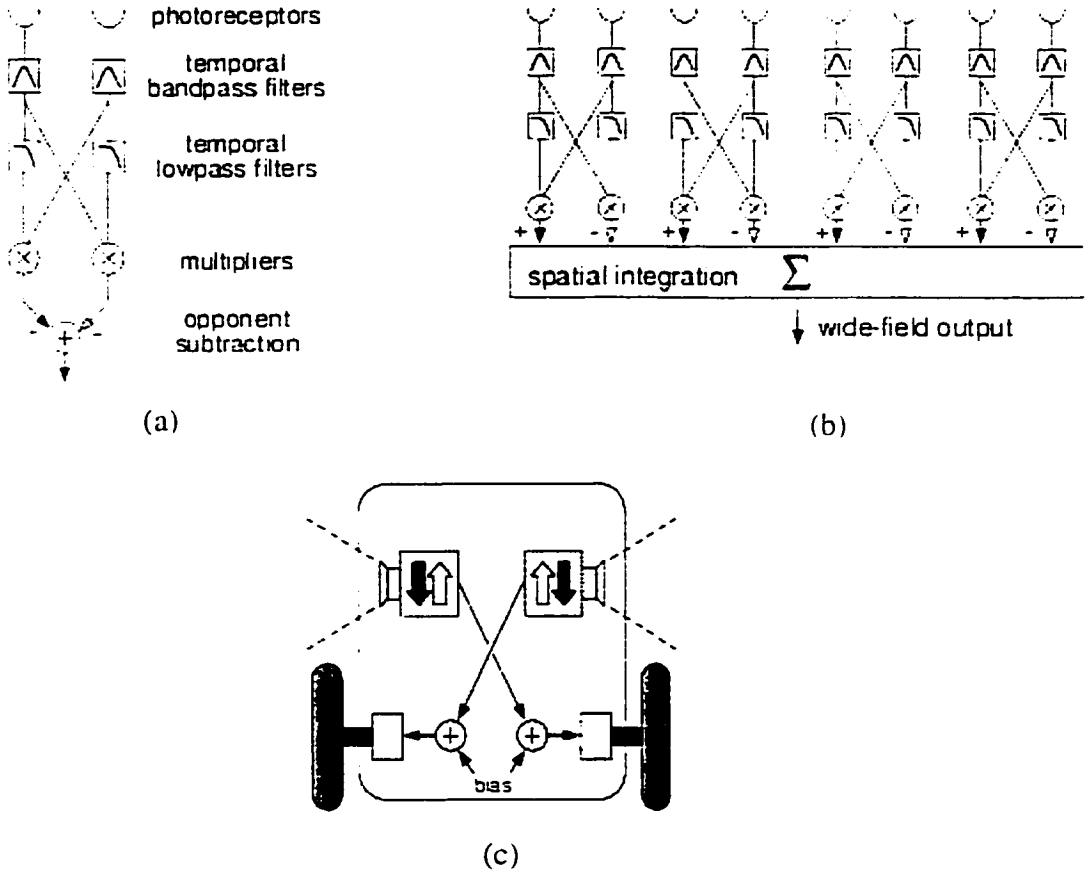


Fig. 2.9 The motion detection approach of Harrison [17]. An elementary motion detector (EMD) is shown in (a), and in (b) a group of EMDs are spatially summed to handle textures. Two side mounted sensors, shown in (c) can be used for obstacle avoidance.

Another example of the motion detection approach was described by Etienne-Cummings [18]. It consists of a two-dimensional array which makes use of foveation by having two regions of different resolution, as shown in Fig. 2.10. At the center of the chip is a 9x9 foveal region where photoreceptors and edge-detection circuits are densely packed. It uses

a resistive network (like in the silicon retina) to realize an edge detector, followed by EMDs to compute the speed and direction of motion of the edges. A potential target's speed is given by the time interval between the disappearance of an edge at a pixel and its re-appearance at a neighboring pixel: the direction is signaled by which neighboring pixel receives the edge. The direction of motion is then used to produce a signal used off-chip to produce an incremental motor adjustment to keep the target centered on the fovea.

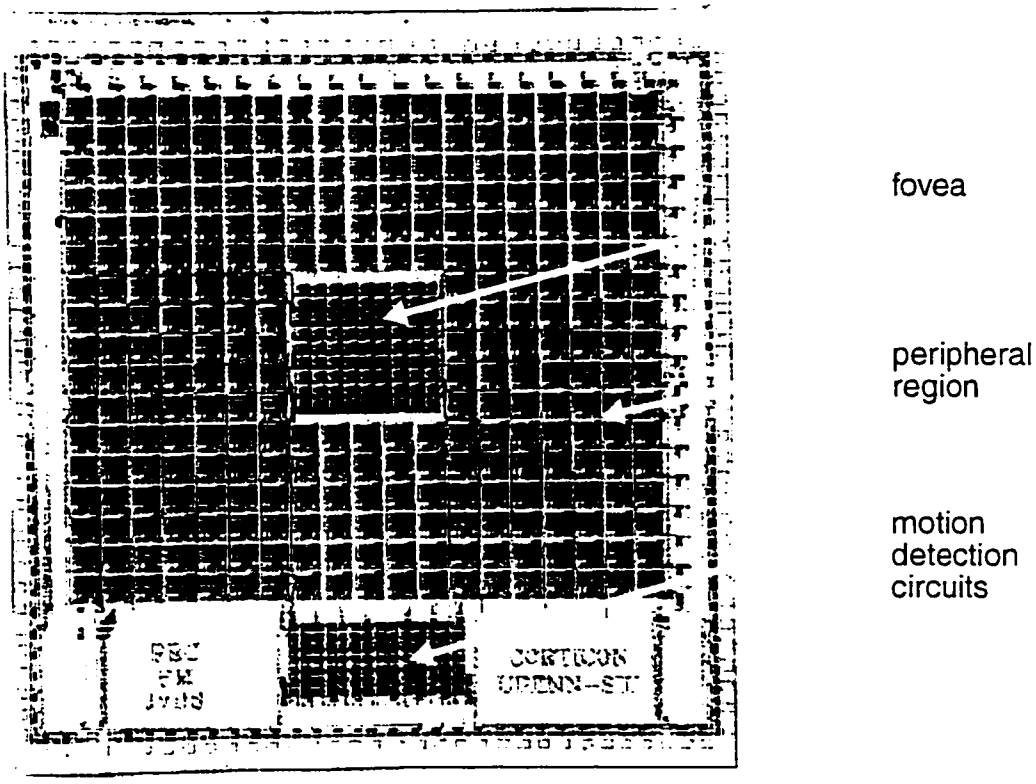


Fig. 2.10 A photomicrograph of a sensory-motor chip which makes use of both motion detection and foveation [18].

Surrounding the fovea is the peripheral region which is used to locate new targets, which in this case are edges. High-resolution imaging is not required. The peripheral cells perform a temporal derivative of the binary edge image, and determine the location of

arriving edges When an edge appears at a pixel, it sends its location to the edge of the array by activating a row and column line. The location of the triggered row (column) is then given by an analog value which is read from a resistive divider. The position can be used off-chip to produce a coarse motor action to adjust the orientation of the system. One of the applications of the chip was a line-following robot.

2.5 Conclusion

Obstacle avoidance in an unstructured environment is a difficult task for an autonomous mobile robot. Traditional approaches to the task have often been inefficient and more suited to simplified environments. Some new ideas and approaches, which have been explored over the last fifteen years by various researchers, were described in this chapter. Some of these ideas have inspired, or are directly incorporated into, the work of this thesis.

First, behavior-based robotics provides an approach which is ideally suited to VLSI sensory-motor systems. This is because it avoids complex data structures and detailed models of the environment. Instead, it emphasizes the particular requirements of the behavior itself, along with the properties of the environment and how the robot interacts with it. It looks for simple ways to more directly couple the sensory-motor system of the robot to the environment.

Secondly, the various smart sensors discussed in this chapter have exhibited some very useful strategies. Perhaps the most important of which, using only simple operations, was present in most of the early sensors. In subsequent sensors, operations such as spatial averaging or summation were shown to be useful in smoothing over some of the high fre-

quency patterns in nature that can lead to unnecessary complexity in the processing. Related to this is the way spatial derivatives and differencing can be used to bring out important details from low frequency backgrounds. Finally, foveation was used by some sensors to prioritize or categorize features according to their position in the visual field.

Despite the usefulness of these strategies, there are still difficulties with achieving an effective obstacle avoidance behavior in an unstructured environment. The first objective of this thesis, as stated in Chapter 1, is to develop a VLSI sensory-motor architecture for an obstacle avoidance task in an environment where there are no simple visual cues to follow. Chapter 3 will describe how the architecture in this thesis achieves that objective by making use of some of the strategies described above in a unique way. In the computer simulation of Chapter 4, the architecture is shown to be an effective way to transform a visual signal to a motor control signal for unstructured environments.

3

System Architecture

In general, the functionality of a signal processing system is determined by its architecture which defines the functional blocks that are used and the way they are interconnected. For systems based on the smart sensor paradigm, the building blocks are generally analog-mixed signal processing operators. Because of imaging and implementation constraints, these operators should be simple in their design and operation, and should use as little space and power as possible. The operators should also be connected in a regular and

scalable manner to reduce the complexity of interconnection wiring and to facilitate the implementation. This chapter presents an architecture for a VLSI sensory-motor system based on the smart sensor paradigm. It is designed to perform a visually-guided obstacle avoidance behavior in an unstructured environment, for typical wheeled robots. The architecture is based on two organizing principles which are derived from a consideration of this behavior, as described in section 3.1. The first organizing principle, described in section 3.2, is a special foveation scheme which is designed to emphasize parts of the visual signal which are relevant to obstacle avoidance. The second organizing principle, described in section 3.3, involves the alignment of sensory and motor maps to more easily convert the visual signal to a motor signal to be used to steer around obstacles. A complete sensory-motor architecture for obstacle avoidance, based on these principles, is described in section 3.4. It is shown to make use of only a small number of computationally simple operators, connected in a regular and hierarchical manner. Because of this, the architecture should lead to a straightforward design and implementation with reasonably sized pixels. By supporting the massively-parallel smart sensor paradigm it could facilitate the realization of system-on-a-chip (SOC) integration for various automatic control systems.

3.1 A Smart Sensor Approach Based on Behavior

3.1.1 An Approach to Suit the Smart Sensor Paradigm

Traditional approaches to achieving visually-guided obstacle avoidance have generally used conventional processing methods which make use of a CPU or DSP connected to an image sensor via an analog-to-digital converter. They make use of algorithms that rely on constructing a detailed, two or three-dimensional model of a scene in memory. Process-

ing these models is often computationally intensive due to the large volume of data.

In contrast, the approach taken in this thesis proceeds within the smart sensor paradigm. This requires the use of simpler algorithms and processing architectures which can be mapped to the type of signal processing supported by this paradigm. The development of such an architecture, as described in this thesis, draws inspiration from the field of behavior-based robotic control, which was described in Section 2.4 of Chapter 2. In the behavior-based approach, emphasis is placed on the definition of behaviors, along with establishing a more direct and complementary coupling between sensing and acting. In a similar way, the sensory-motor system in this thesis is explicitly designed to extract behaviorally relevant information from a scene in order to avoid obstacles. The starting point is a clear definition of an obstacle avoidance behavior.

3.1.2 A Simple Definition of An Obstacle Avoidance Behavior

For an obstacle avoidance behavior, a robot should steer its body away from any stimulus which might signal that a significant object is in its pathway. As it moves, it should be orienting itself towards an open part of the pathway in front of it. The basic requirements are:

1) real-world obstacles must be detected and located relative to the robot's body,

2) the robot should steer to the left or right around the obstacle in a reactive manner.

3.1.3 Use of Foveation

If, as the first requirement suggests, only relative locations are needed, then actual positions in a coordinate frame need not be computed. The first requirement also suggests

that obstacles need not be recognized or even resolved with much precision; they only need to be detected. In addition, resolving power does not even have to be the same for all parts of the scene. More distant parts of the scene do not need high resolution because in the distance only the detection of large objects is needed to make steering decisions. Space-variant or foveated sensing schemes have been shown to have these qualities, in both biological [10] and engineered systems [23]. Thus, the first organizing principle for the sensory-motor system in this thesis is that a foveated sensing scheme should be used. The particular scheme proposed in this thesis is referred to as post-receptor foveation, and is described in section 3.2.

3.1.4 Alignment of Sensory and Motor Maps

The second requirement of the obstacle avoidance behavior suggests that for a typical wheeled robot, only a left/right orientation decision is needed to avoid an obstacle. This greatly simplifies the task of converting the visual signal to a motor signal. The fact that only a left/right decision needs to be taken can be exploited by aligning the map of the visual field (as represented on the sensor), with the center of the robot's body. If more obstacles are on the left of the visual field, then they are to the left of the robot's body and the robot should turn to the right. This is a simple alignment of a sensory map and a motor map, and is the second organizing principle for the architecture. An enhanced version of this alignment is described in detail in section 3.3.

3.2 Post-Receptor Foveation

3.2.1 Hierarchical Averaging

As mentioned in chapter 2, there have been several sensor architectures reported in the literature which make use of foveation schemes [24][25][26]. All of these schemes vary the sizes and the placement of individual receptors in the array, e.g. log polar arrangements. However, for a VLSI smart sensor system, an architecture based on a homogeneous array of uniform pixels is easier to design and extend. The pixel can be designed like a standard cell with a constant pitch, and can be arranged to abut with common power rails and interconnections. This would reduce the number of unique cells that must be designed and could make it easier to scale or modify the system. Thus, a foveation scheme based on a homogeneous array of uniform receptors is proposed in this chapter. In this scheme, called post-receptor foveation, the variation in resolution is achieved by spatially averaging the signals from the photoreceptors. The basic structure is shown in Fig. 3.1. Each photoreceptor produces a signal, like a voltage or a current, which represents the intensity of the light which projects onto its area from the scene. The average of the signals from a 'field' of four adjacent photoreceptors is called a level-1 average. Following this, the average of four adjacent level-1 averages is then called a level-2 average and so on, in a hierarchical manner. In this way, each level defines a set of fields which cover the array of photoreceptors. The field size at level-1 is the smallest. At each successively higher level the field size increases by a factor of four. A larger field can be said to contain or overlap the smaller fields which contribute to its value. The hierarchical simplicity of the scheme resembles the pyramid data structures often used in image segmentation [27], rather than following a polar distribution.

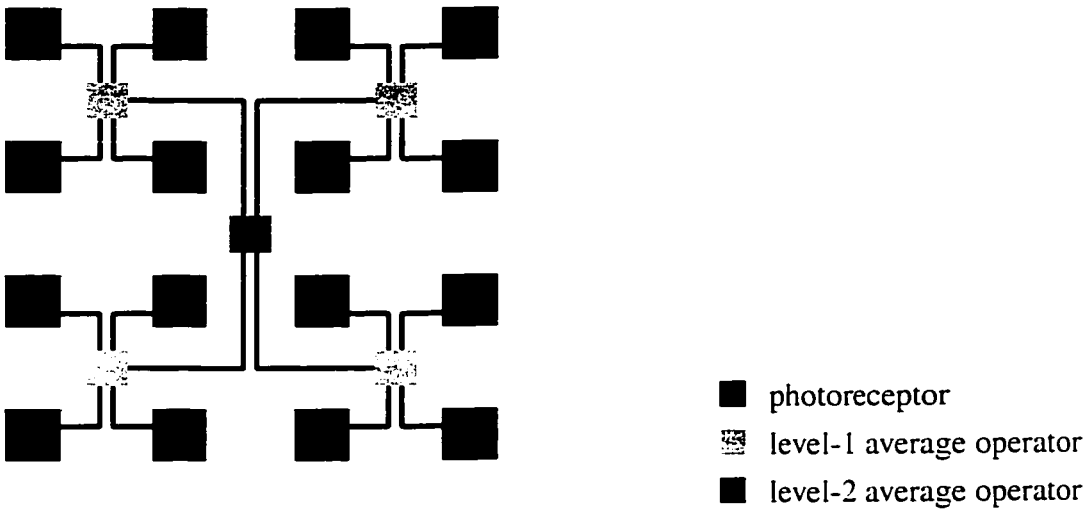


Fig. 3.1 Basic structure for the hierarchical averaging of a homogeneous array of receptors.

3.2.2 Defining the Fovea

In general, the fovea refers to the area on a sensor in which the highest resolution is available for processing. It is usually surrounded by an area which has a decreasing resolution away from its center. In the post-receptor foveation scheme just described, the available resolutions are determined by the interconnection structure of the array. The resolution that can be obtained by accessing the level-1 averages, or fields, is four times the resolution that can be obtained by accessing the level-2 fields. A fovea can be defined over an area where the smallest fields (eg. level-1) are used for processing. A particular example is shown in Fig. 3.2, where the fovea is defined to be in the shaded corner area where the difference between two level-1 fields, e.g. a spatial derivative, is computed. Operations on larger fields, like the level-2 spatial derivative, can overlap the fovea but are not considered part of the fovea.

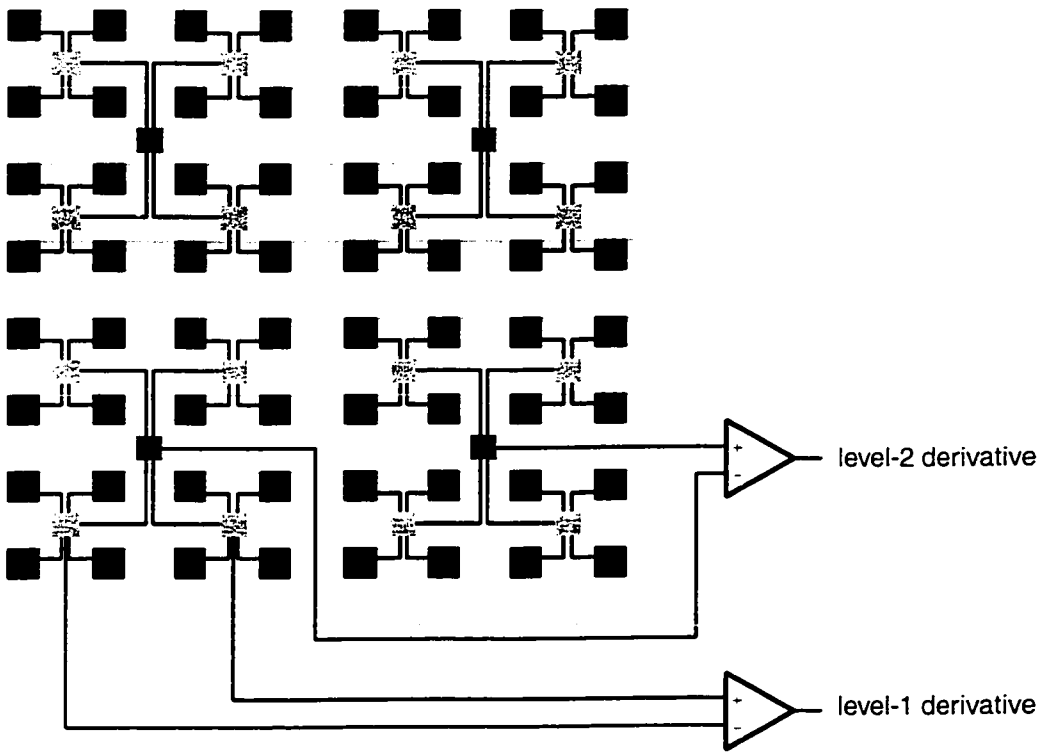
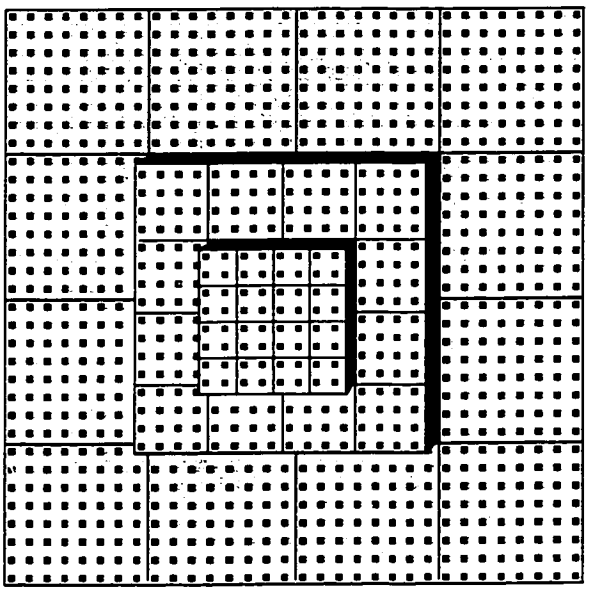
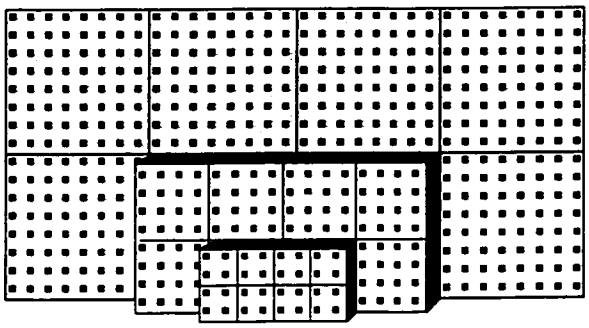


Fig. 3.2 Defining a fovea on an 8x8 array. The fovea consists of the four level-1 fields in the unshaded corner. Each of these fields contains four photoreceptors, whereas the level-2 derivative is performed on fields which contain 16 photoreceptors each.

In this way, the fovea can be placed anywhere on the array and can assume different shapes. This is similar to the situation in some birds which are known to have an elongated fovea (to align with the horizon), or even multiple foveas as in the case of the owl [28]. These arrangements can facilitate the extraction of critical information from situations which are common to the organism. Using a homogeneous array as a substrate, a diagram of some possible configurations is shown in Fig. 3.3 (not showing the average operators and interconnections). The sensor in Fig. 3.3 (a) has the fovea in the center of a 32x32 array array while in (b) the fovea is at the bottom of a 32x16 array.



(a) A centered fovea on a three-level 32x32 array



(b) A bottom-edge fovea on a rectangular 32x16 array

Fig. 3.3 Positioning the fovea. The fovea may be positioned anywhere on an array of receptors, including the center, as in (a), or on the perimeter, as in (b).

3.2.3 Advantages for Applications in Unstructured Environments

This type of foveation offers a couple of important advantages when it is applied in realistic, unstructured environments. First, the hierarchical averaging produces a type of low-pass filtering of the image which is increasingly coarse away from the fovea. This can make complex objects like bushes or rocks appear more like a block of uniform intensity,

facilitating their detection based on intensity differences. Emphasizing parts of the visual signal with lower spatial frequencies helps to filter out distracting details which could complicate the process. A second advantage is that hierarchical averaging leads to a convenient foveated framework for other types of operations that can be incorporated at any level of the hierarchy. For example, the density of the receptor array remains uniformly high, making it possible to add an operator like a temporal derivative, to each pixel as shown in Fig. 3.4. This can be used to distinguish objects based on differences in their texture and the amount of 'flicker' they produce, within the foveated framework. As another example, a simple motion detector could be integrated with each photoreceptor to detect and measure optical flow in the horizontal and vertical directions across the array. Combining multiple cues in this way can help to disambiguate obstacles in unstructured environments.

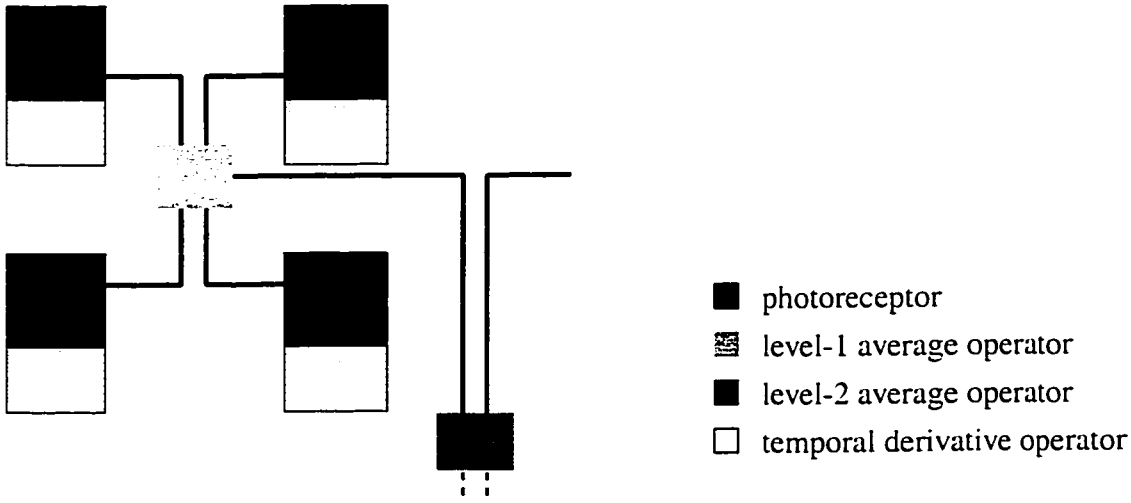


Fig. 3.4 Augmenting the basic structure with a per-pixel temporal derivative operator.

3.3 The Alignment of Sensory and Motor Maps

3.3.1 Sensory-Motor Transformations

A sensory-motor transformation is a process by which a sensory signal is converted to a motor signal. Traditional approaches often express this as a coordinate transformation from the coordinate system of a three-dimensional world model to the robot's local coordinate system, in order to specify its orientation. It is difficult to implement these types of coordinate transformations in the smart sensor paradigm. As mentioned in chapter 2, there have been some strategies based on optical flow, but these have also proven to be computationally intensive [29][30] making it challenging to use optical flow in smart sensor architectures [17][31].

The scheme proposed in this thesis, to perform sensory-motor transformations, is based on the fact that a representation of the visual field is constantly present across the photoreceptor array. This is referred to as a sensory map. If the only degree of freedom to be controlled is a left/right steering action, then in a similar way, the motor map can be defined as a 'left control signal', $c_L(t)$, and a 'right control signal', $c_R(t)$. The left and right control signals are derived from the left and right sides of the visual field respectively, aligning the two maps as shown in Fig. 3.5.

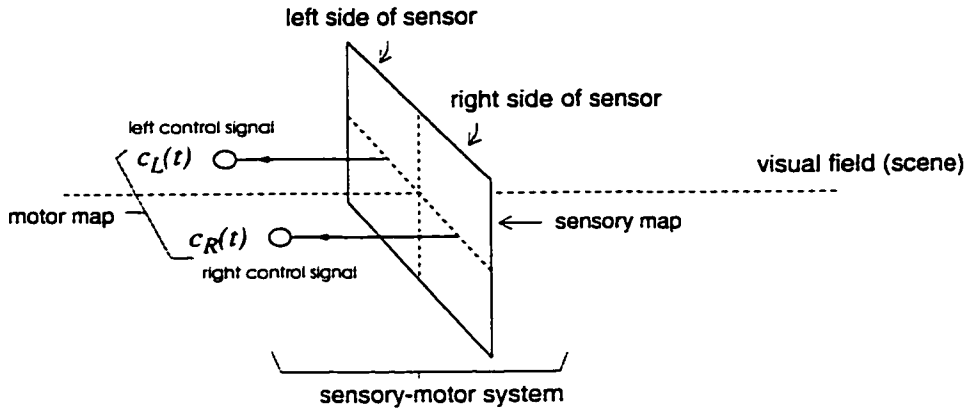


Fig. 3.5 A sensory-motor system viewed as an alignment of a sensory map and a motor map.

This alignment scheme is motivated by similar sensory-motor mechanisms thought to exist in the superior colliculus, as discussed in section 2.2 of Chapter 2. Sensory and motor maps in the superior colliculus overlap and are aligned so that activity in a particular region of the sensory map will trigger activity in the motor map, leading to a rapid orientation toward that region of the visual field. A smart sensor is an ideal substrate for this type of processing because the map of the visual field is always present across its photoreceptor array. Later, in section 3.3.3, an enhancement to this alignment scheme will be presented. In a similar way, the enhancement scheme is based on mechanisms thought to exist in the posterior parietal cortex, as was also discussed in section 2.2 of Chapter 2. This region of the brain receives both visual signals and eye position signals, and produces a response which is modulated by the current position of the eye [13]. It has been postulated that this modulation is used to transform eye-centered coordinates to body-centered coordinates by taking into account any shift in gaze angle. In a similar way, the system discussed in this thesis will use weights, or gain factors, which depend on the vertical gaze angle of the sensor.

3.3.2 A Simple Configuration for a Robot with a VLSI Sensory-Motor System

The left and right control signals should reflect the degree to which the left and right sides of the visual field are obstructed. The post-receptor foveation scheme of section 3.2 can be used for this purpose. As mentioned in Chapter 2, the most frequently cited advantage of foveation is the ability to combine high visual acuity (near the optical axis) with a large field of view, while using fewer receptors and reducing the amount of data to be processed. In addition to this, however, Ballard [32] showed that foveation could be used to establish a frame of reference centered on an object in the scene. The frame of reference is referred to as the 'fixation frame'. Its center is the 'fixation point' which is the place in the scene where the optical axis, or fovea, is directed as shown in Fig. 3.6. An observer can shift the fixation point to any object in the scene to suit its current information gathering needs. Other objects in the scene can be located relative to the fixation point in an imprecise but efficient manner, due in part to the decrease in resolution away from the fixation point.

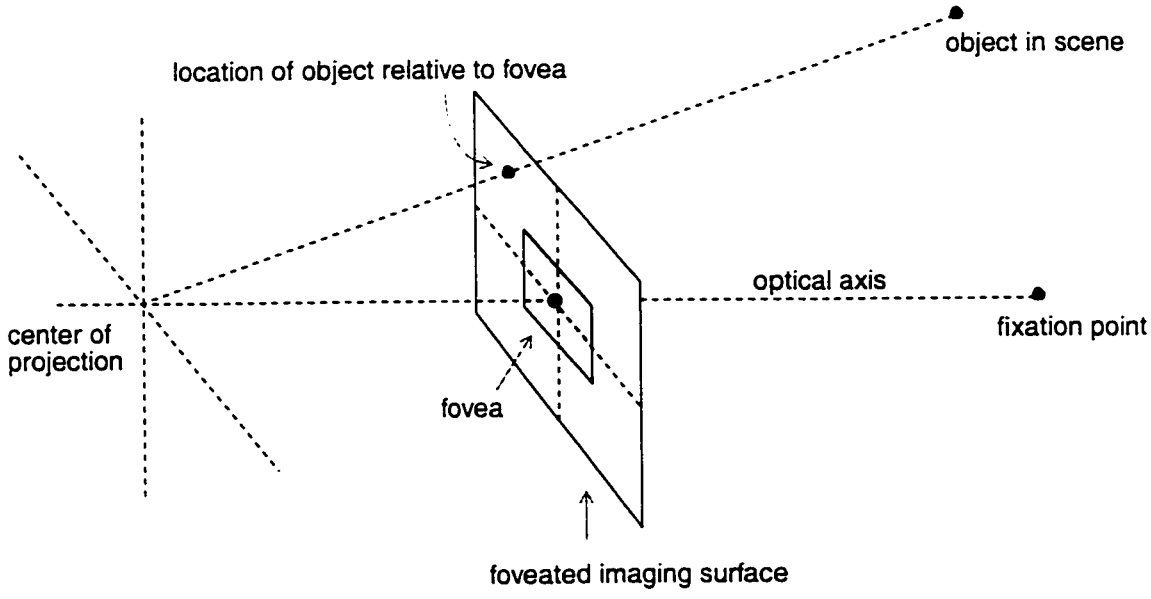
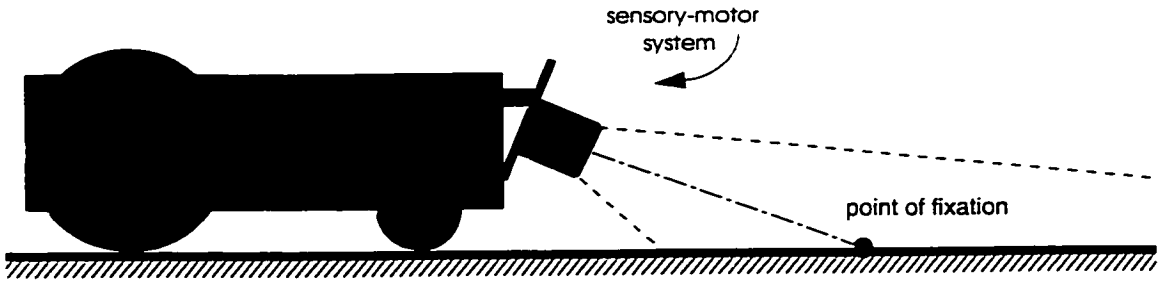
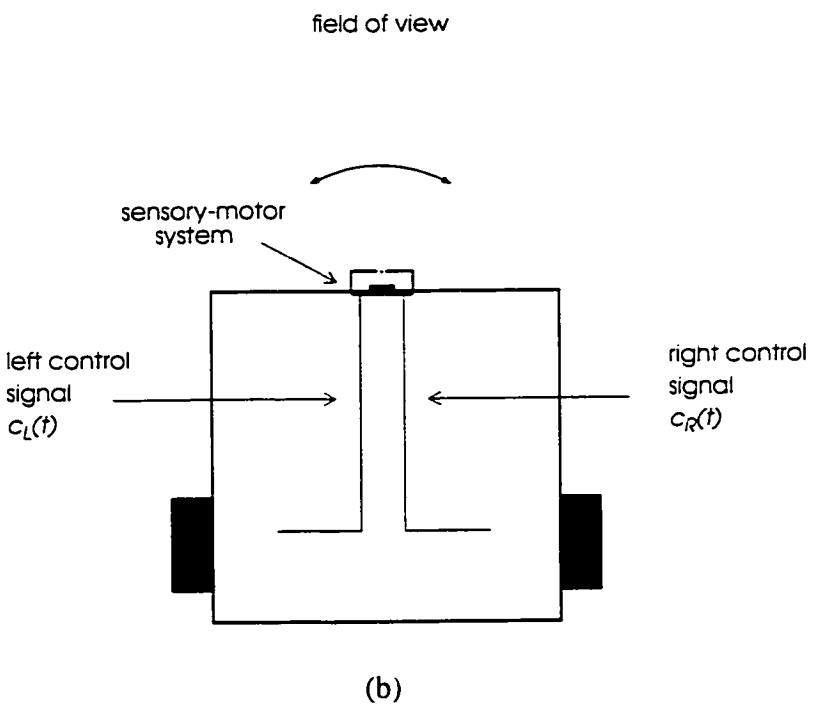


Fig. 3.6 The fixation frame. The location of objects relative to the fovea gives a quick approximation of their location in the scene, relative to the fixation point (any point of interest).

To illustrate how this can be used, an example configuration for an autonomous robot with a VLSI sensory-motor system is shown in Fig. 3.7. The sensory-motor system is mounted on the front of the robot with the center of the sensor aligned with the midline of the body. Making use of the strategy in [20], the sensor is angled toward the pathway, as shown in Fig. 3.7 (a), so that the point of fixation is immediately in front of the robot where the path is assumed to be 'open'. If all potential obstacles are on the ground plane (or near to it), then the more distant they are from the robot, the higher up they will appear on the image plane. Obstacles can then be located relative to the open path in front of the robot by using this measure of depth along with a determination of whether they lie more to the left or right of the point of fixation. Since the center of the sensor is aligned with the midline of the robot's body, the point of fixation and the visual field are also aligned with the body. This means that if an obstacle lies more on the left side of the sensor, then the robot should steer to the right. By using spatial derivatives and other visual cues to determine where the open pathway is limited by obstacles, a continuous signal from each side of the sensor can be produced, the magnitude of which indicates the degree to which the side is obstructed. Thus, a left/right control signal can be obtained through this simplified transformation.



(a)



(b)

Fig. 3.7 An autonomous robot with a VLSI sensory-motor system mounted on the front. Shown in (a) is the side view of a robot with a sensory-motor system angled toward the ground. Shown in (b) is a top view of robot showing the left and right motor signals from the sensory-motor system.

Fig. 3.8 shows an example 16x8 sensor array configured in this manner. The sensor's width is double the height in order to increase the field of view laterally and to accomodate the rectangular foveation. The fovea is at the bottom of the array in order to establish a fixation frame centered on the open pathway immediately in front of the robot. This allows the rest of the height of the sensor to be used to map scene depth. The extent of the open pathway on each side are expressed as continuous left and right control signals, $c_L(t)$ and $c_R(t)$.

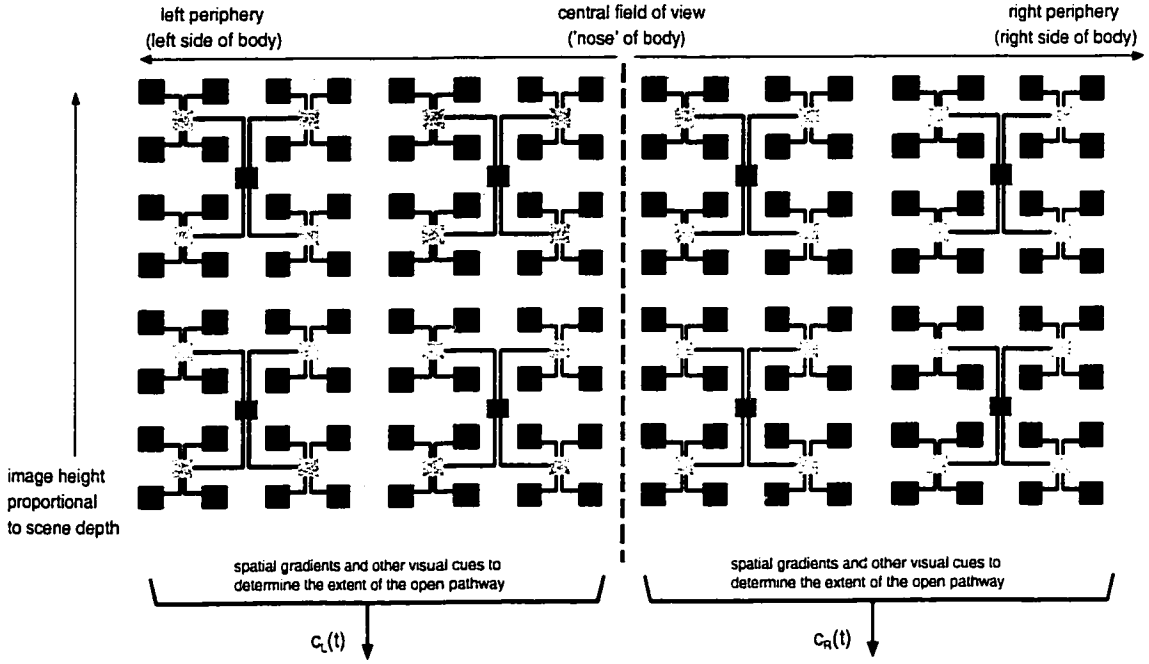
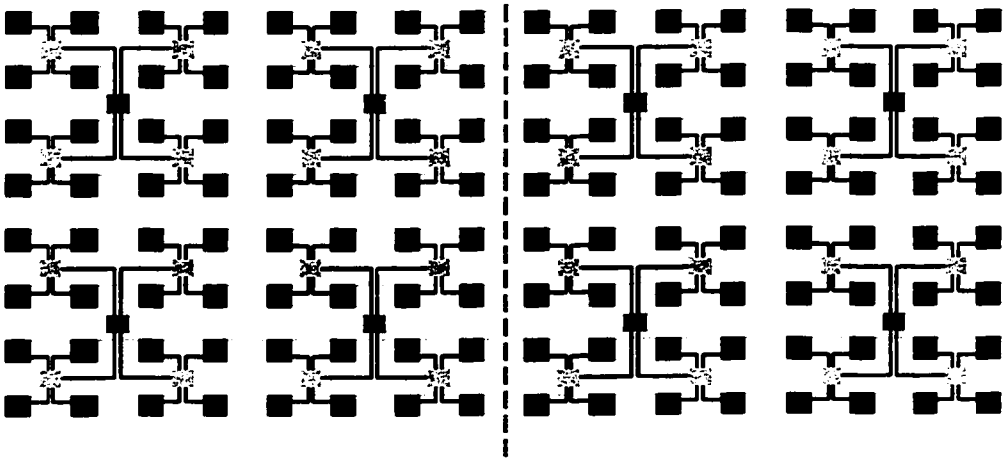


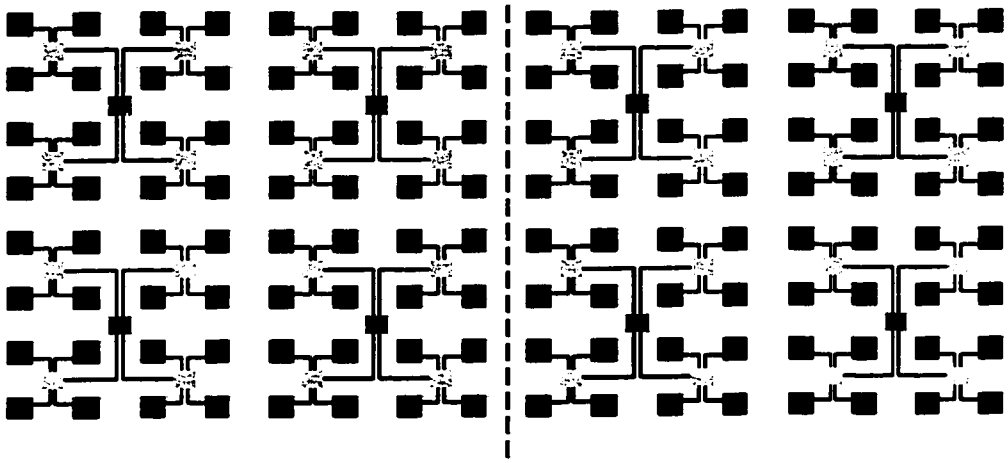
Fig. 3.8 A rectangular array divided into left and right halves. The fovea (unshaded) is aligned with the midline of the body to facilitate a left/right steering decision.

3.3.3 An Enhancement to Map Alignment

The simple association of the left and right halves of a sensor with the left and right halves of the robot's body and the visual field can be further enhanced. Central regions of the sensor can be associated with the 'nose' of the robot and peripheral regions can be associated with the sides. This refinement can be used to add a weighting scheme to the obstacle detection cue such that objects which are more immediate and closer to the nose will elicit a stronger signal for the robot to turn away from them. Central, diagonal and side (or peripheral) regions may be distinguished to refine the response of the system. Fig. 3.9 shows the diagonal fields that would be used in a computation (like a spatial derivative), where the smallest fields consist of four receptors, and the next level fields have 16 receptors. A spatial derivative across the smaller fields can have a larger weight, or gain factor, than for the next level field because it would indicate a more immediate obstacle. In section 3.4, below, a description of a complete architecture shows more clearly how these weights are used.



(a) Foveal, or level-1 fields which consist of four receptors each.



(b) Level-2 fields consist of 16 receptors each and overlap the fovea.

Fig. 3.9 The diagonal region of an example 16x8 array is composed of four level-1 diagonal fields, shown in (a), and four level-2 diagonal fields, shown in (b).

3.4 A Complete Sensory-Motor Architecture

The post-receptor foveation scheme of section 3.2 and the enhanced map alignment scheme of section 3.3 can be combined to produce a complete sensory-motor system for obstacle avoidance. The architecture is depicted in Fig. 3.10. For clarity, only the right side of the system is shown and not all level-1 derivatives are depicted. Comparator symbols are used to show the spatial derivatives which are computed for the three regions, center, diagonal and side.

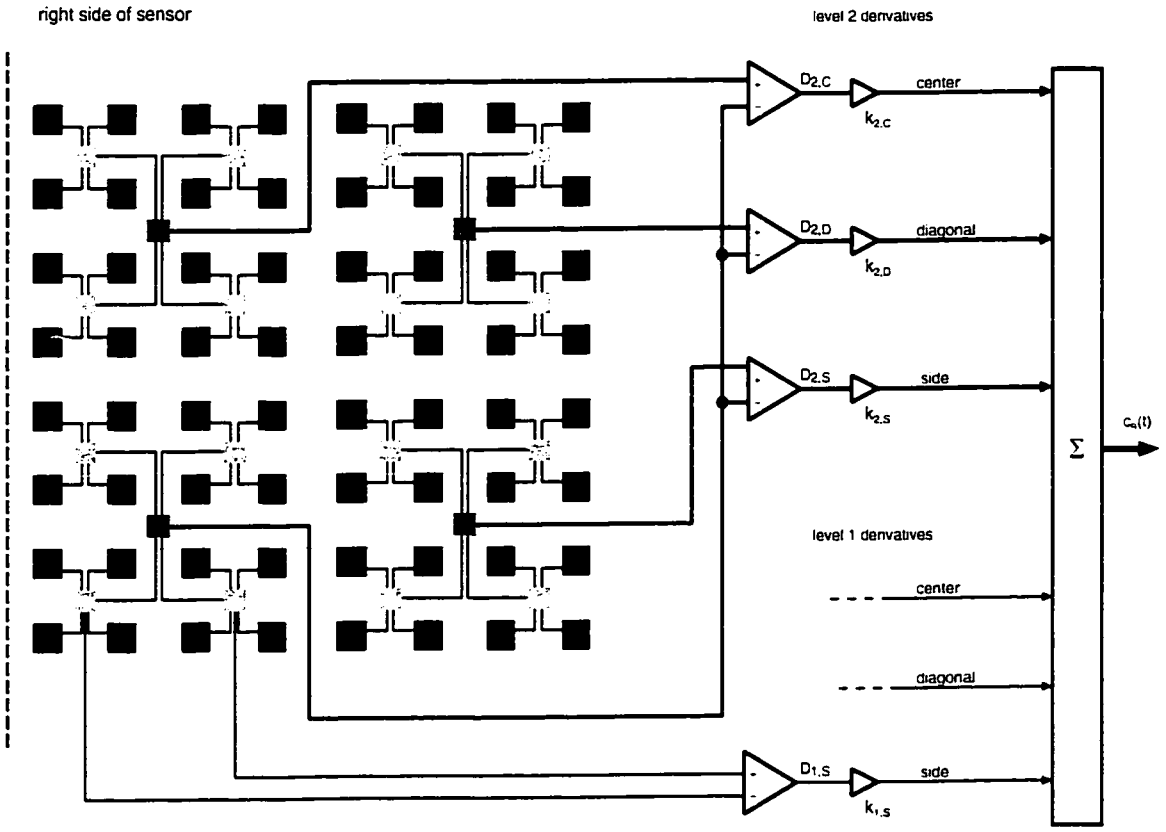


Fig. 3.10 The right side of a sensory-motor architecture. The structure comprises hierarchical averaging, spatial derivative operations with scale factors, and a summation of all levels.

The center derivative can signal an object in the robot's face, while diagonal and side derivatives are progressively less frontal. The gain factors following the derivatives should reflect this. For example, the second level gain factor for the center is shown as k_{2C} and for the diagonal and side as k_{2D} and k_{2S}. In most cases they should follow the general relationship: $k_{2C} > k_{2D} > k_{2S}$. These factors can be related to the shape of the robot's body (narrow/wide) in order to determine more precise ratios. Derivatives for lower levels will signify more immediate obstacles so that if k_{iX} is used to refer to all gain factors at level i, then it should be that $k_{iX} > k_{(i+1)X}$, for $i > 1$. The angle at which the sensor is mounted on the robot (the vertical gaze angle) can be used to determine more precise inter-level ratios for these constants. Finally, by summing the scaled derivatives a continuous signal is produced to indicate the extent of the free pathway. The difference between left and right signals can then be used to steer the robot.

3.5 Summary

This chapter presented an architecture for a VLSI sensory-motor system, based on the smart sensor paradigm, to perform visually-guided obstacle avoidance in an unstructured environment. The architecture made use of two organizing principles: (a) a special foveation scheme designed to emphasize parts of the visual signal which are relevant to obstacle avoidance, and (b) the alignment of sensory and motor maps to more easily convert the visual signal to a control signal to be used to steer around obstacles.

The post-receptor foveation scheme results in a structure which is regular and hierarchical, and which uses uniformly spaced pixels of identical size. In summary, the advantages related to using post-receptor foveation are:

- pixels are like standard cells with a constant pitch, designed to abut with common power rails and interconnections across the array
- hierarchical averaging makes all levels of resolution available for processing, supporting spatiotemporal operations down to the pixel level
- hierarchical averaging results in a regular interconnection structure which can be extended easily for larger arrays
- large fields of receptors can help to deal with variance in receptor response, such as noise, device variation, and receptor failure.

The alignment of sensory and motor maps enables a sensory-motor transformation in a simple and computationally efficient manner. In summary, the advantages related to using this scheme are:

- a sensory-motor transformation is achieved without the use of explicit coordinates or optical flow computations
- the close coupling of sensory and motor maps enables rapid orientation behavior

The architecture makes use of four basic signal processing operators, all of which have standard analog VLSI circuit implementations:

1) an average computation
2) a difference computation
3) a scaling
4) a summation

The fact that the operators are computationally simple could lead to smaller pixel sizes and higher fill-factors. The next chapter presents an analysis and two computer simulations of the architecture, and then chapter 5 presents example circuits for the four operators.

4

An Evaluation of the Architecture through Analysis and Simulation

The previous chapter described a signal processing architecture for a sensory-motor system based on the smart sensor paradigm. This chapter describes a simple mathematical model of the architecture. The model, presented in Section 4.1, is used as the basis for computer simulation of the architecture. Two simulations are described. The first simulation, presented in Section 4.2, is designed to verify that the architecture is able to extract a

useful signal for motor control from a still image. The second simulation, presented in Section 4.3, is designed to verify that the architecture which was presented in Chapter 3, can actually perform an obstacle avoidance behavior in an unstructured environment. It uses a three-dimensional virtual environment and is based on a simple imaging model described in this chapter. The simulation is also used to determine the minimum pixel size for a potential implementation.

4.1 A Mathematical Model of the Architecture for Computer Simulation

4.1.1 Mathematical Expressions for the Sensory-Motor Architecture

This section identifies a set of mathematical expressions to model the sensory-motor architecture in order to simulate it. Fig. 4.1 shows the right half of a homogeneous array of photoreceptors. As described in Chapter 3, the array is partitioned into levels where each level consists of four fields. Each field belongs to one of the regions: 'path', 'center', 'diagonal', 'side'.

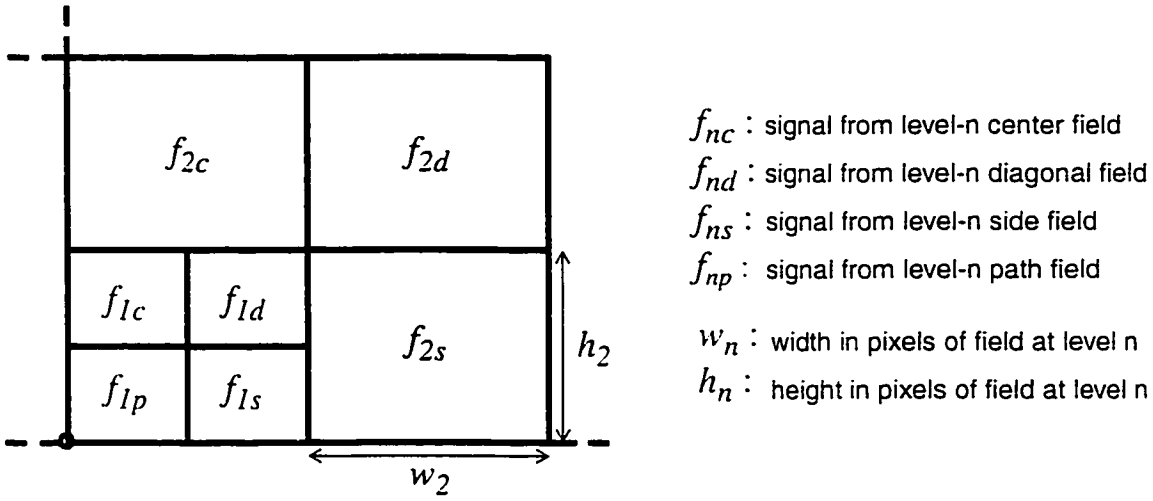


Fig. 4.1 Right half of the array showing the notation used to refer to the four fields at each level.

The value of each field is the average of the magnitudes of the signals from its constituent pixels. Using the notation shown in Fig. 4.1, this can be expressed as the summation of the magnitudes for all of the field's pixels, divided by the number of pixels in the field. If $I(x,y,t)$ denotes the magnitude of the signal which represents the intensity of light on the pixel at position x,y and at time t, then the expressions for the signals of the center, diagonal, and side fields at the nth level are:

$$f_{nc}(t)= \frac{1}{w_n \cdot h_n} \cdot \sum_{1}^{w_n} \sum_{h_n}^{2h_n} I(x, y, t), \quad n \geq 1 \tag{1}$$

$$f_{nd}(t)= \frac{1}{w_n \cdot h_n} \cdot \sum_{w_n}^{2w_n} \sum_{h_n}^{2h_n} I(x, y, t), \quad n \geq 1 \tag{2}$$

$$f_{ns}(t)= \frac{1}{w_n \cdot h_n} \cdot \sum_{w_n}^{2w_n} \sum_{1}^{h_n} I(x, y, t). \quad n \geq 1 \tag{3}$$

where w_n and h_n are the width and height in pixels of the fields at level n. The expression for the signal for the 'path' field at level-1 is:

$$f_{1p}(t)= \frac{1}{w_1 \cdot h_1} \cdot \sum_{1}^{w_1} \sum_{1}^{h_1} I(x, y, t), \tag{4}$$

and for subsequent levels, the signals from the previous levels are used:

$$f_{np}(t)= \frac{1}{2 \cdot 2} \cdot (f_{(n-1)p}(t) + f_{(n-1)c}(t) + f_{(n-1)d}(t) + f_{(n-1)s}(t)), \quad n > 1. \tag{5}$$

Expressions (1) to (3) are used to compute the differences between the values of the path field and the center, side, and diagonal fields, eg. the spatial derivatives, as described in Section 3.4. The left and right control signals, $c_L(t)$ and $c_R(t)$, can be expressed as the summations of all of these spatial derivatives for all levels. Each derivative is multiplied by a

weight which depends on its level and its region. These weights are: K_{nc}, K_{nd}, K_{ns}. As an example, the right control signal, $c_R(t)$, is:

$$c_R(t) = \sum_{1}^{L} K_{nc} \cdot [f_{nc}(t) - f_{np}(t)] + \sum_{1}^{L} K_{nd} \cdot [f_{nd}(t) - f_{np}(t)] + \sum_{1}^{L} K_{ns} \cdot [f_{ns}(t) - f_{np}(t)] \quad (6)$$

where L is the highest level. The signal can vary between 0 and some value c_{MAX}. A value of 0 occurs when all the derivatives are 0, indicating that the path is completely free of obstacles. A value of c_{MAX} occurs when the sums of the derivatives are a maximum, strongly indicating the presence of an obstructed path.

The functionality of the architecture described in Chapter 3 does not go beyond the left and right control signals. For the purposes of a complete obstacle avoidance simulation, the control signals had to be used to determine the change in a robot's orientation. The change in orientation angle, denoted $\Delta\alpha(t)$, is proportional to the difference between the left and right control signals, with a scaling factor K_α, to convert to an angular measure like radians:

$$\Delta\alpha(t) = K_\alpha \cdot [c_L(t) - c_R(t)] \quad \text{radians.} \quad (7)$$

The signals c_{MAX}, $c_L(t)$ and $c_R(t)$ would be either voltage or current signals in an actual smart sensor implementation. A constant like K_α is a coefficient for signal conversion from voltage or current to radians, giving it units like radians/V or radians/A. On a real robot this would represent the conversion of the steering signal to an actual change in direction.

4.1.2 From Architecture to Algorithm for Computer Simulation

For a computer simulation, the expressions in the above model had to be applied to every frame in a discrete-time sequence of images. Each frame was processed in the same way, using the algorithm, **process_frame,** shown in the flowchart of Fig. 4.2. The first step involves computing the average for every field at every level of foveation, as described in equations (1)-(5) in the previous section. The second step involves computing the left and right control signals, in the same way as in equation (6). In the third step, the change in orientation angle is computed as in equation (7).

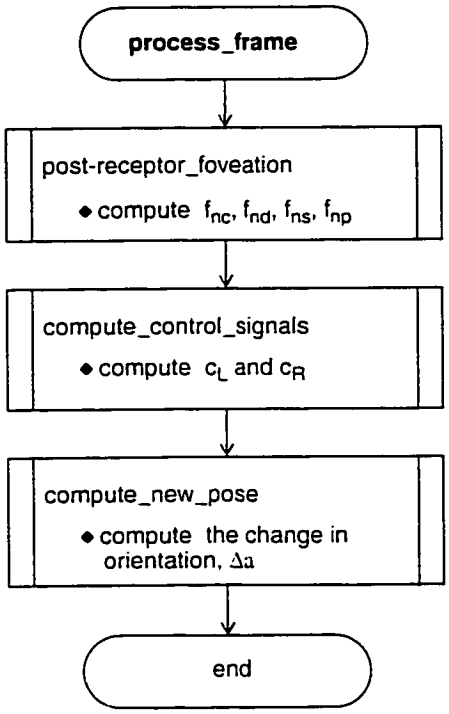


Fig. 4.2 Flowchart of the algorithm used to process each frame in a simulation. The first step is the post-receptor foveation which is described by equations (1)-(5) and is shown in detail in the flowchart of Fig. 4.3.

The **post_receptor_foveation** step is described in more detail in the flowchart of Fig. 4.3. The most important thing about the algorithm is that it consists of simple computations which must be done repetitively across the array. This type of algorithm is ideally suited to the smart sensor paradigm because the computations are simple and may be performed at the pixel level in parallel.

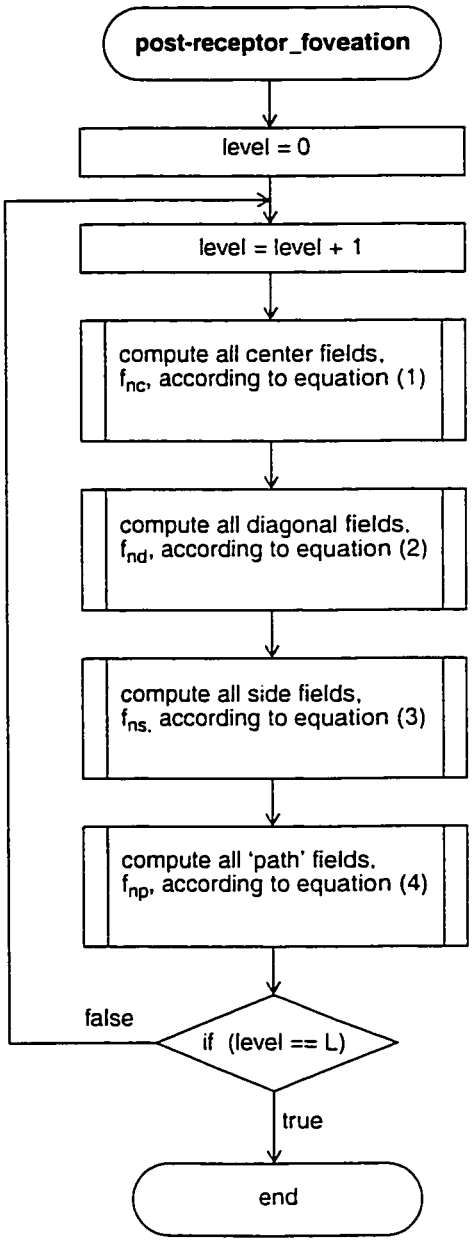


Fig. 4.3 Flowchart of the algorithm used to compute the values for all of the fields for all levels of foveation. It is the first step in the flowchart of Fig. 4.2.

4.2 Computer Simulation with a Still Image

4.2.1 Purpose of the Simulation

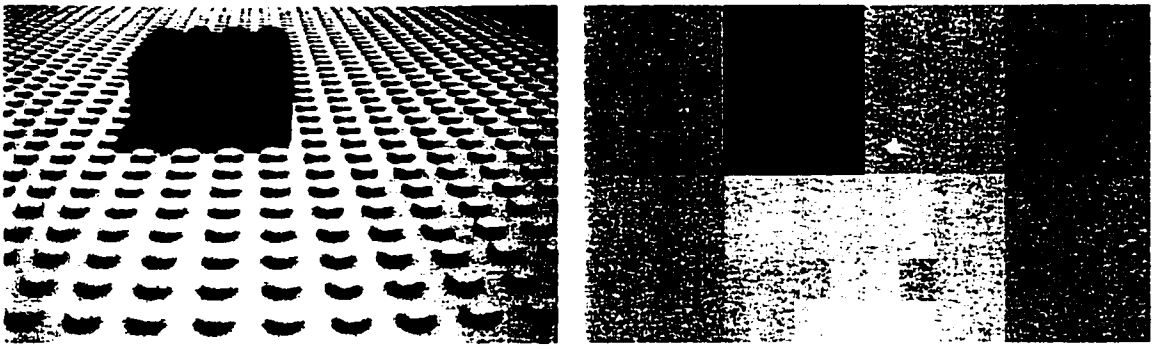
The simulation with a still image was intended to be a preliminary verification of the effectiveness of the architecture. The purpose was to determine if it would be possible to derive reasonable control signals, $c_R(t)$ and $c_L(t)$, from a 2D signal array acquired from a simple test scene. Positive results were needed to proceed with confidence to the more elaborate simulation described in Section 4.3.

4.2.2 Methods Used for the Simulation

For the simulation, a simple test scene was configured with a dark cube resting on a textured surface. A grey-level image of the scene was subsampled down to a resolution of 16x8 to match the architecture shown in Fig. 3.14 of Section 3.4 in Chapter 3, and to show that the architecture can be effective at such a low resolution. The test image is shown in Fig. 4.4 (a). Pixel values ranged from 0 to 256. The post-receptor foveation algorithm of Fig. 4.3, on page 53, was applied to the image. For each level of foveation, the differences between the path field and the surrounding center, diagonal and side fields were determined, as in equation (6), but with unity weights. These differences, or spatial derivatives, were then plotted using Matlab to form a type of gradient map.

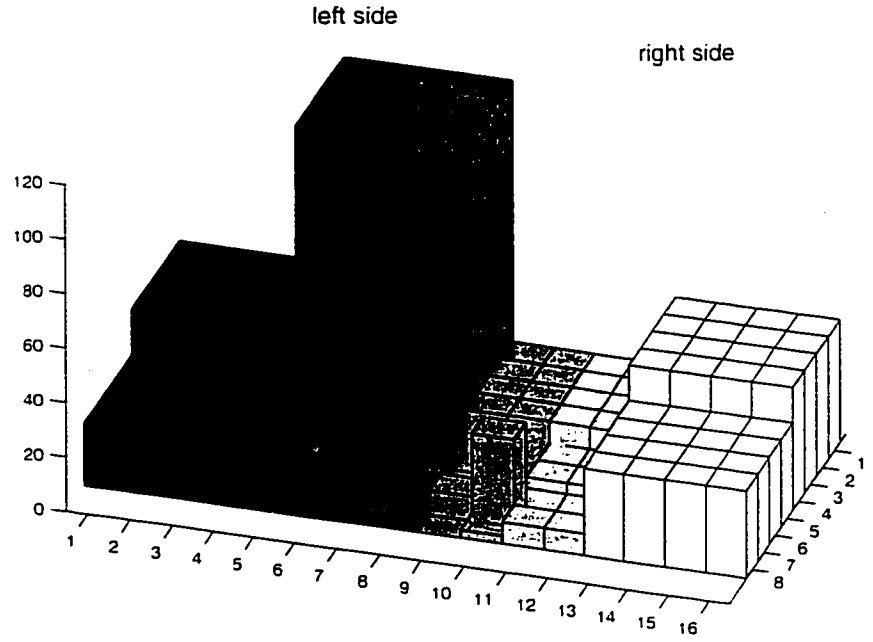
4.2.3 Results

The result of applying the foveation step to the test image is shown in Fig. 4.4 (b) and the plot of the differences is shown in (c). The plot itself is only for visualization. The values for the differences have been distributed according to the level and region they



(a) Scene with simple obstacle on textured surface (b) Foveation applied to the image in (a)

left side:			
$f_{3c}(t) - f_{3p}(t) = 116$	$f_{2c}(t) - f_{2p}(t) = 6$	$f_{1c}(t) - f_{1p}(t) = 15$	
$f_{3d}(t) - f_{3p}(t) = 40$	$f_{2d}(t) - f_{2p}(t) = 3$	$f_{1d}(t) - f_{1p}(t) = 22$	$c_L(t) = 238$
$f_{3s}(t) - f_{3p}(t) = 23$	$f_{2s}(t) - f_{2p}(t) = 9$	$f_{1s}(t) - f_{1p}(t) = 4$	
right side:			
$f_{3c}(t) - f_{3p}(t) = 20$	$f_{2c}(t) - f_{2p}(t) = 4$	$f_{1c}(t) - f_{1p}(t) = 9$	
$f_{3d}(t) - f_{3p}(t) = 45$	$f_{2d}(t) - f_{2p}(t) = 12$	$f_{1d}(t) - f_{1p}(t) = 32$	$c_R(t) = 165$
$f_{3s}(t) - f_{3p}(t) = 32$	$f_{2s}(t) - f_{2p}(t) = 8$	$f_{1s}(t) - f_{1p}(t) = 3$	



(c) Plot of the differences which were derived from the foveated image in (b)

Fig. 4.4 Test image and simulation results clearly indicating an obstacle on the left.

belong to in the foveation scheme. The actual differences, specified in equation (6), were calculated for all three levels and are shown in the figure along with the values for the left and right control signals. The large peak on the left side of the plot signifies the presence of the black cube unambiguously. The result indicates that it would be possible to use the architecture to derive reasonable control signals, $c_R(t)$ and $c_L(t)$, from unstructured scenes. A clear decision to turn to the right can be derived from the plot, as can be seen by the difference between $c_R(t)$ and $c_L(t)$. It is interesting to note that the textured surface had no effect on the results. The foveated image shows how the surface variations are averaged away. They did not create a distraction for the process. Nor were they necessary for the detection of the obstacle.

4.3 Computer Simulation with a Virtual Environment

4.3.1 Purpose of the Virtual Environment Simulation

The purpose of the second simulation was to verify if the architecture which was presented in Chapter 3, would lead to a system which could perform an obstacle avoidance behavior in an unstructured environment. The simulation was also intended to verify that the architecture is suitable for a smart sensor implementation by determining parameters such as pixel size, based on an imaging model for the sensor.

4.3.2 Methods used for the Virtual Environment Simulation

To simulate an unstructured environment, a computer program was developed using the OpenGL graphics library. The program made use of techniques in three-dimensional computer graphics to create a virtual environment. The virtual environment could be made to look realistic by applying patterns to the floor and other objects. The view of the virtual environment was designed to represent how it would appear from the view of a sensory-motor system mounted on the front of a robot. The program, hereafter referred to as the simulator, was designed with an interactive user-interface, shown in Fig. 4.5.

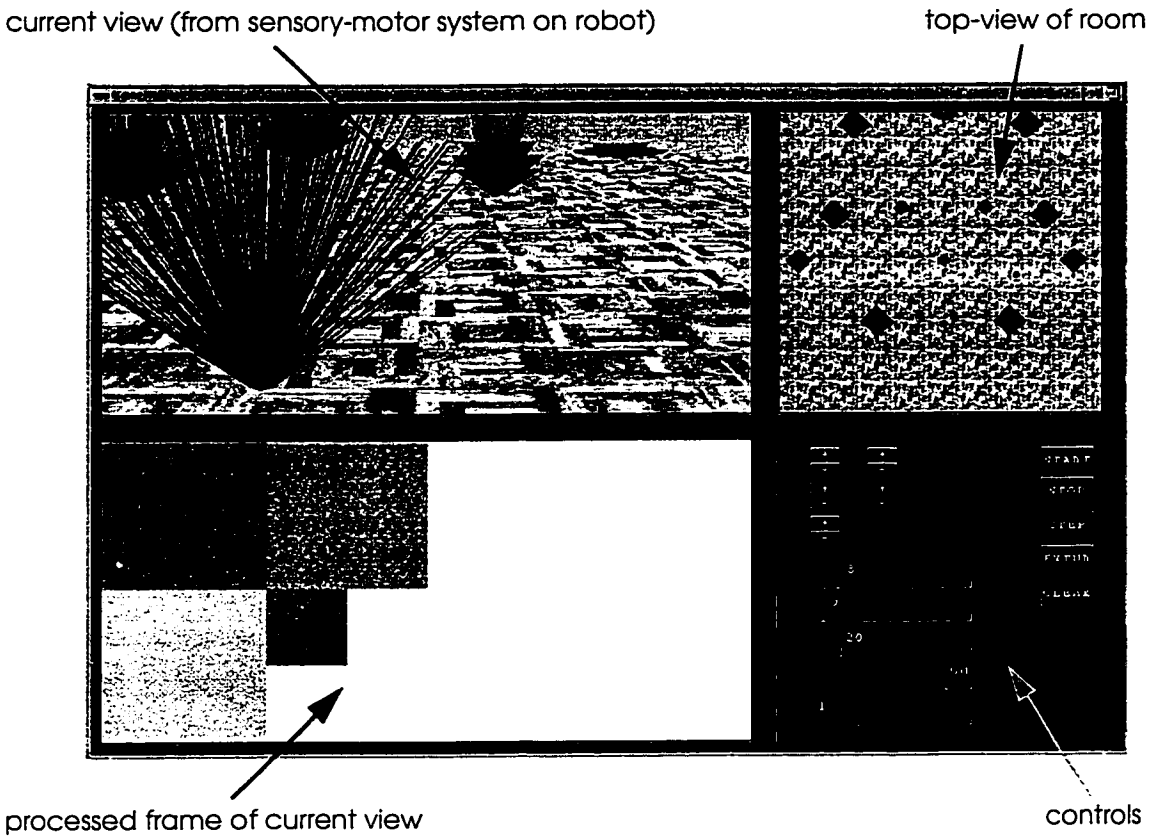


Fig. 4.5 User-interface for the simulator. The top left panel contains the current (perspective) view of the room, the top right panel contains the top view of the room and the bottom left panel contains the foveated image of the current view.

As shown in Fig. 4.5, the top-left panel of the simulator's interface displays the current (perspective) view of an example virtual environment as it would be seen if one were to look from the point of view of the sensory-motor system on the robot. In this example, the virtual environment is a room with a hardwood floor and obstacles which resemble small bushes. The top right panel displays a top-view of the room, and will show a trace of the path of the robot through the room as the simulation proceeds. The bottom-left panel contains a foveated image of the current view, as created by the post-receptor foveation algorithm in Fig. 4.3. The lower right panel contains a set of controls to set the parameters for a simulation run.

The parameters for the simulation include the angle at which the sensor is mounted on the robot, the height of the mounting, the focal length of the imaging system, and the size of the sensor itself. These parameters are part of a simple imaging model which the simulator used to recreate the mapping of a realistic scene onto an actual sensor. The details of this model are described in the next section.

4.3.2.1 An Imaging Model for the Sensory-Motor System

The current (perspective) view from the robot, as displayed in the top left panel of the simulator, had to correspond closely to what would actually be acquired by a physical sensor. To achieve this, a simple model based on the classic pinhole camera scheme, as shown in Fig. 4.6, was used to model the image formation process for the simulation. It was used because it is simple and can be easily simulated using standard 3D computer graphics.

The pinhole camera scheme assumes that light rays pass through a pinhole sized aperture at the front of the imaging system or camera. The geometry of image formation in such an arrangement is closely approximated by perspective projection. The aperture corresponds to the center of projection, C, as shown in Fig. 4.6. The center of projection is the origin of the camera coordinate system denoted by x_c, y_c, z_c. The optical axis is the line of sight which passes through the center of projection and is aligned with the z_c axis. The image plane is at a distance f, the focal length, behind the center of projection and the projected image is inverted. This is the case in an actual camera. However, the image plane can be thought of as being in front of the center of projection as a mirrored plane, to simplify the derivation of the mathematical model. A rectangular sensor array can be thought

of as being in the image plane with its center aligned with the origin of the plane. Image plane coordinates are expressed as u,v. Perspective projection is accomplished by extending a straight line from each point in the scene to the center of projection. Each straight line corresponds to a ray of light which would be emitted or reflected by the scene point. The intersection of the line with the image plane determines where each scene point is mapped. The figure also shows how the view volume and the field-of-view angles, ϕ_x, ϕ_y, are defined by the size of the sensor array and the focal length.

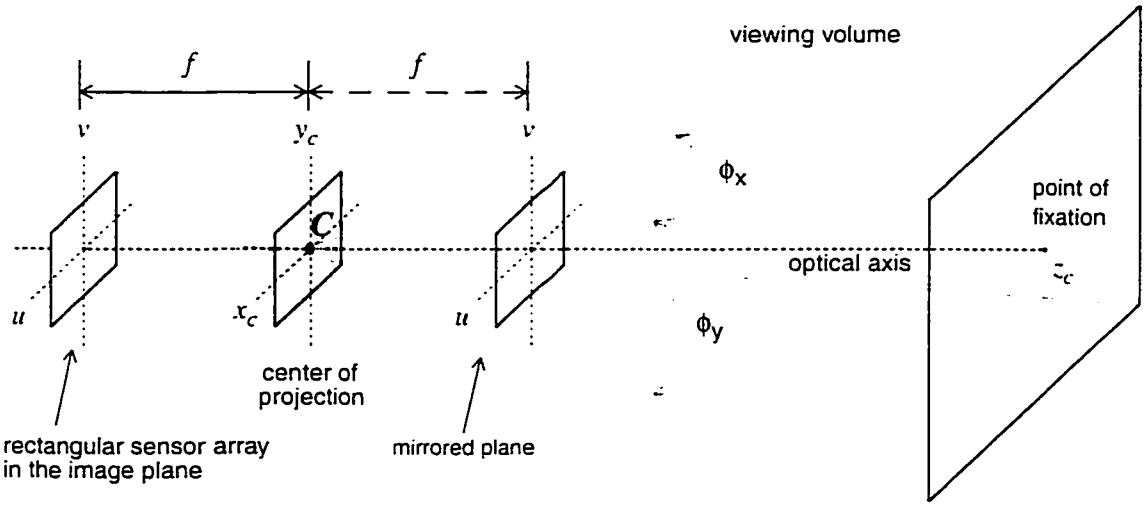


Fig. 4.6 Pinhole camera scheme for image acquisition with a rectangular sensor array. Perspective projection can be used to map the sizes of objects in the scene onto the surface of the sensor which lies in the image plane u,v. For the purposes of illustration the focal length, f, is shown much larger than it would actually be.

With this scheme, objects in the scene can be mapped onto the surface of the sensor array, in the u,v plane, by using perspective projection. For obstacle avoidance, it can be assumed that there is some minimum size for objects in the scene, below which they are not significant. Only objects which are larger than the minimum size are considered poten-

tial obstacles. By mapping a minimum sized object onto the sensor array it is possible to determine how large the smallest fields of the post-receptor foveation should be in order to detect it. The smallest fields are the level-1 fields and they are square so that their length and width can be denoted by σ_1. They provide the finest resolution used to compute spatial derivatives around the point of fixation. The fields should be kept small enough so as not to lose important features in the scene. On the other hand, the larger they are, the larger the photoreceptor that can be used in each, resulting in stronger signals.

To determine the value of σ_1, it is assumed that only obstacles or features which are more than 1/5th of the robot's body width are to be regarded as potential obstructions to its progress. The most difficult to detect objects will be ones which are very low or do not rise above the ground plane. A hole in the ground is a good example. Fig. 4.7 shows how a circular hole with diameter $w/5$, where w is the width of the robot, would be projected onto the sensor using the perspective projection model. The sensor's orientation in the y-z plane is denoted by the vertical gaze angle, θ, between the optical axis and a line perpendicular to the ground plane. The height of the sensor from the ground plane, h, is measured along a line from the center of projection (aperture) to the ground plane. The distance, d_{min}, denotes the distance between the sensor and the intersection of the optical axis with the ground plane. The arrangement in the figure has the optical axis at the bottom of the sensor where the fovea is, as described in Fig. 3.14 of chapter 3.

In the system shown in Fig. 4.7, the ground-plane is projected onto the sensor resulting in a rectangle with a width of u_o and a height of v_o covering the hole. Because of the gaze angle θ, the height will be compressed more than the width when the ground plane is mapped to the sensor.

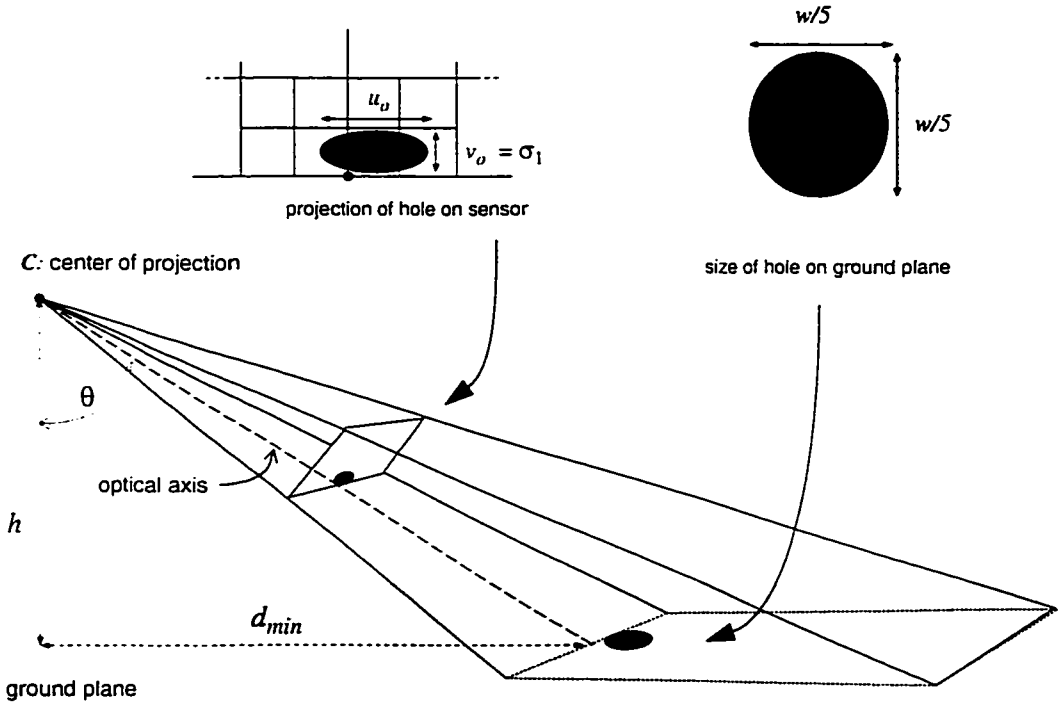


Fig. 4.7 A circular hole in the ground is used to represent the smallest feature that must be detected. The hole is mapped to the sensor using perspective projection, taking into account the height, h, of the center of projection from the ground plane, and the gaze angle, θ. Proportions have been exaggerated to show the parameters with clarity.

Thus, the size of the level-1 field, σ_1, is determined according to the compressed height, v_o. The dimensions u_o and v_o can be expressed in terms of the vertical gaze angle, θ, and the parameters of the configuration, such as f, the focal length, w and h (the derivation for these expressions is presented in Appendix A):

$$u_o = f \cdot \frac{w}{5h} \cdot \cos(\theta) \ , \ v_o = \frac{f \cdot \frac{w}{5 \cdot h}}{(\tan(\theta))^2 + \frac{w}{5 \cdot h} \cdot \tan(\theta) + 1} \quad \text{(mm)} \qquad (8)$$

As a practical example, for a sensory-motor system mounted at a height of h = 50mm, on a robot of width w = 50mm, at a vertical gaze angle of θ = 60 degrees, and with a focal length f = 3mm, then:

$$u_o = 300\mu \quad , v_o = 140\mu$$

Taking the smaller of the two, 140μ, as the dimension for the smallest square field at level 1, σ_1, then a level-2 field has a side of 280μ and a level-3 field has a side of 560μ. Thus, a three level arrangement would have a total width of u_T = 2240μ and a total height of v_T = 1120μ, or a total size of 2.24mm x 1.12mm. The total size and the focal length can be used to calculate the horizontal field-of-view angle, ϕ_x. The result is around 37 degrees, which is similar to that of a typical 35mm camera with a 50mm lens. These are reasonable numbers for an implementation of the system. Thus, the analysis suggests that such a system is feasible if a level-1 field can have the dimensions 140μ x 140μ. Of course, for a much larger robot the dimensions will be larger. For example, if w = 500mm, then σ_1 = 800μ in order to be sensitive to an obstacle of 100mm width.

4.3.2.2 A Simplified Kinematic Model for the Simulation

The imaging model of the previous sub-section was used to map parts of the scene to the sensor plane in a realistic manner. After the imaging mapping and the signal processing of the architecture, the left and right control signals are generated to make a steering decision. To simulate this in a realistic manner, a model of a wheeled robot is used to verify that the steering decisions in the simulation are realistic. For this purpose, a wheeled robot with a differential steering system is used. In such a system, the left and right drive wheels are independently powered and their relative velocities determine the steering angle of the robot. Fig. 4.8 shows a top view of the drive wheels of such a robot. The robot is located on the ground plane at position, (x,y) and with an orientation angle, α.

The simulation proceeds on a frame by frame basis and each frame of the simulation is assumed to represent the passing of 1/60th of a second to match the video frame rate of

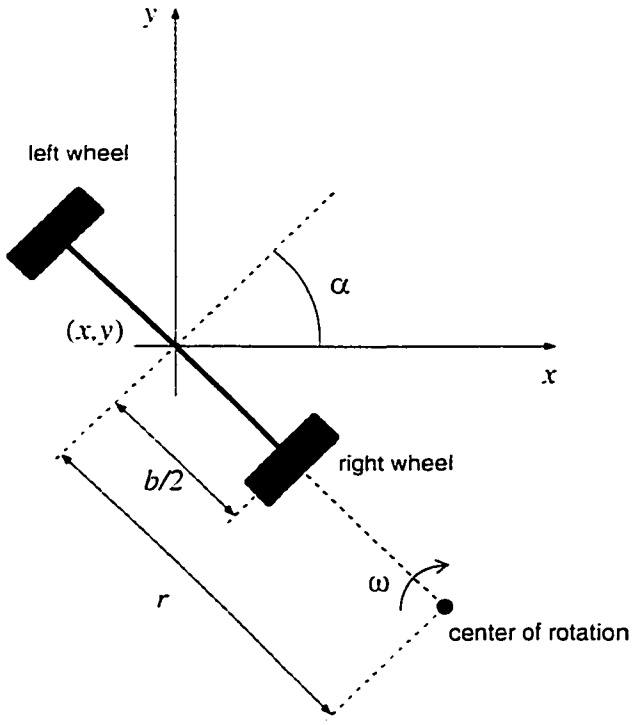


Fig. 4.8 Top view of a robot's drive wheels with the parameters for a kinematic model.

a typical video camera. Assuming that a simulated robot of 50cm width and length is moving at an average speed of about 1.8 m/second, the speed of a brisk walk, then the simulator should move forward by 3cm per frame. The maximum change in orientation per frame should be determined to suit these parameters. Referring to Fig. 4.8, the left and right wheels will have velocities in the direction of roll, given by the products of the turning radius, r (adjusted by half the axle length, b), and the angular velocity ω:

$$v_R = \omega\left(r + \frac{b}{2}\right) \quad , \quad v_L = \omega\left(r - \frac{b}{2}\right) \tag{9}$$

The angular velocity about the center of rotation can then be expressed as:

$$\omega = \frac{(v_R - v_L)}{b} \tag{10}$$

This may then be expressed in terms of the orientation angle, α:

$$\frac{d\alpha}{dt} = \frac{(v_R - v_L)}{b} \tag{11}$$

Integrating both sides, and assuming constant velocities and an initial orientation angle, α_0:

$$\alpha(t) = \frac{(v_R - v_L)}{b} t + \alpha_o . \tag{12}$$

This is the forward kinematics expression for the orientation angle, α, which can be determined at any time, t, given the wheel velocities. For a Δt, the change in orientation angle is simply:

$$\Delta\alpha = \frac{\Delta t}{b} \cdot (v_R - v_L) \quad . \tag{13}$$

For a robot with a width of 50cm, b will be 25cm. The time difference, Δt, will be 1/60th of a second. That yields:

$$\Delta\alpha = \frac{1}{15} \cdot (v_R - v_L) \qquad \text{(m/s)} \tag{14}$$

Assume that the maximum achievable velocity difference over this Δt is around 2m/s. The maximum change in orientation per frame will be around 0.133 radians or about 8^o. Consider how the change in orientation angle was previously given by equation (7) in Section 4.1.1:

$$\Delta\alpha(t) = K_\alpha \cdot [c_L(t) - c_R(t)] \quad \text{radians.}$$

where K_α is a coefficient for the conversion of the difference between left and right control signals to radians. Substituting equation (7) into equation (13) yields:

$$K_\alpha \cdot [c_L(t) - c_R(t)] = \frac{\Delta t}{b} \cdot (v_R - v_L) \leq 0.133 \quad \text{radians.} \tag{15}$$

Equation (15) is used to determine K_α. The exact value of K_α depends on the range of the difference between the control signals. It should be set such that the maximum change in orientation is less than 0.133 radians or 8^o.

4.3.2.3 Procedure of the Virtual Environment Simulation

A complete procedure of the virtual environment simulation is shown in the flow-chart of Fig. 4.9. The first step, **create_room**, creates a geometric model of a virtual environment, such as a room, in three-dimensional coordinates. The next step, **set_imaging_geometry,** sets the imaging parameters like the focal length and the field of view, to simulate how the scene would be projected onto a sensor array of a particular size, using the imaging model presented above. After that, **set_view** establishes the initial view of the room which is determined by the position and orientation of the robot, and the angle at which the sensory-motor system is mounted on it. When the simulation loop starts, **draw_room** draws the current view of the room in the top-left panel, and then **process_frame** uses the current view of the room to create the foveated image and to determine the direction the robot should turn, according to the algorithm back in Fig. 4.2. The amount of forward motion for the robot is also determined at this step. After this, **update_view** sets the new view of the room according to the new orientation and position of the robot. The simulation loop then begins again by drawing the new current view, and the cycle continues.

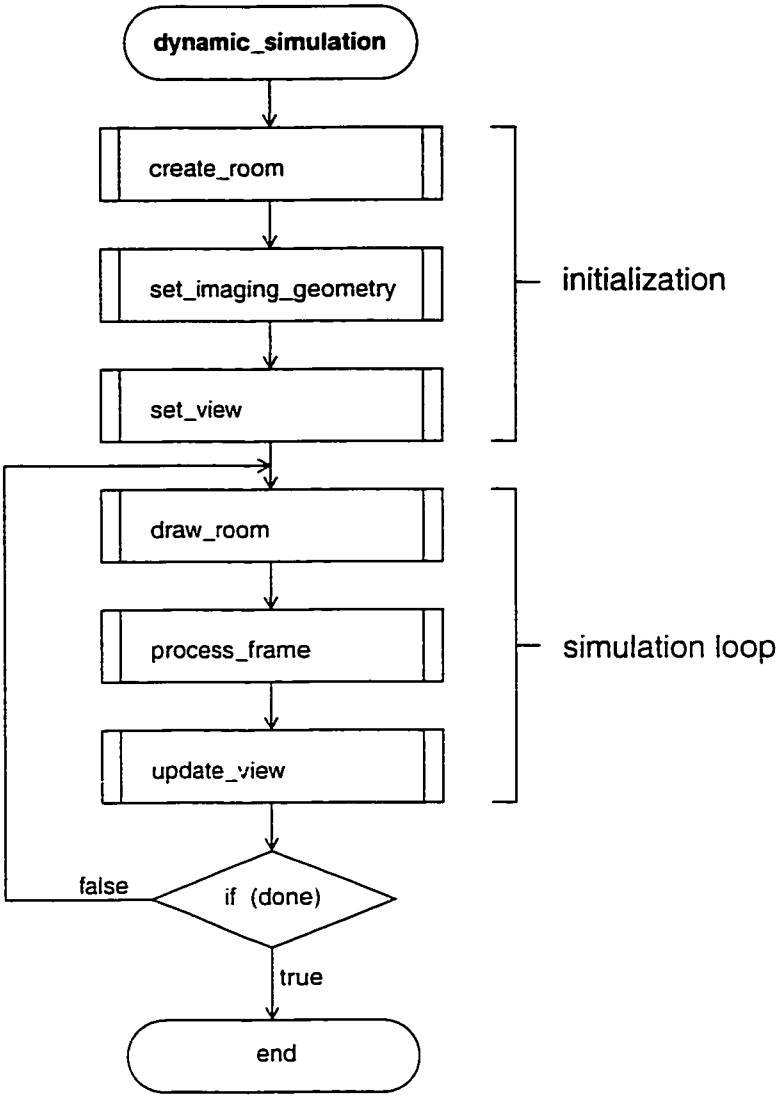


Fig. 4.9 Procedure of the Virtual Environment Simulation

4.3.3 Example of Simulation and the Results

This section describes the settings for the parameters of the simulator and presents the results from several simulation runs. For each run of the simulator, the four imaging parameters were set to the values shown in Table 1.

Table 1:

parameter	value	description
f	3 mm	focal length of imaging system
u_T	2 mm	total width of sensor array
θ	60°	vertical gaze angle
h	50cm	height of sensor from ground plane
w	50cm	width of robot body

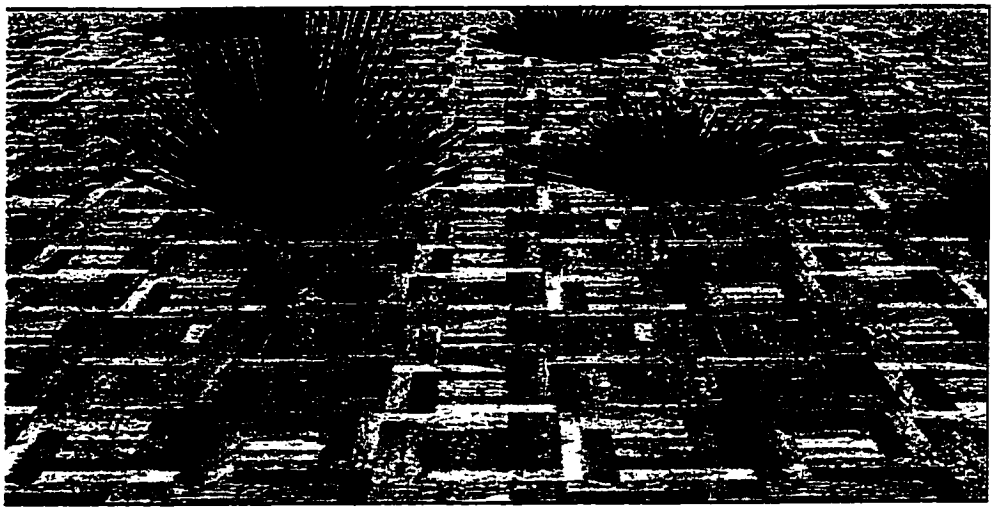
The last parameter, w, the width of the robot body is included as an imaging parameter because it is used to determine the size of the smallest obstacle to be detected as $w/5$.

In addition to the imaging parameters, the weighting scheme for a three level system had to be set in accord with the relationship $k_{iC} > k_{iD} > k_{iS}$ for i = 1,2,3. The values used in the simulator are shown in Table 2.

Table 2:

level 1	level 2	level 3
$K_{1C} = 4.0$	$K_{2C} = 2.5$	$K_{3C} = 1.0$
$K_{1D} = 3.5$	$K_{2D} = 2.0$	$K_{3D} = 0.5$
$K_{1S} = 3.0$	$K_{2S} = 1.5$	$K_{3S} = 0.5$

The scene chosen for the simulations consists of a 5m x 5m room with a typical hardwood floor pattern. The pattern can be replaced with a flat color or other variations. Two types of obstacles were spread out across the floor. The first type was a small green bush and the second type was a circular hole with a 10cm diameter. The bushes were designed to simulate irregular, natural obstacles without well-defined contours. They ranged in size from 10cm to 50cm in diameter. Fig. 4.10 (a) shows how both obstacles appear when using the imaging parameters of Table 1. Fig. 4.10 (b) shows how sensitive the system is to any change in these parameters. A change of only 10^{o} in vertical gaze angle and 10mm in the height of the sensor above the ground plane accounts for the difference in appearance between the two images.



(a) View obtained with $\theta = 60^{o}$ and h = 50cm.

(b) View obtained with $\theta = 70^{o}$ and h = 40cm.

Fig. 4.10 Two views from the simulator using different settings for vertical gaze angle and the height of the sensor above the ground plane.

In particular, the dark hole obstacle is less distorted in Fig. 4.10 (a) in which the imaging parameters of Table 1 are used. These parameters were also used in the analysis of Section 4.3.2.1 to predict that the smallest field size (of level-1 fields) should be approximately 140μ x 140μ, with a sensor width of approximately 2mm, to detect a hole of 10cm diameter. Fig. 4.11 shows how the simulator confirms this prediction. With the sensor width set to 2mm, the foveated image shows that the smallest fields are just able to give a strong indication of the presence of the hole.

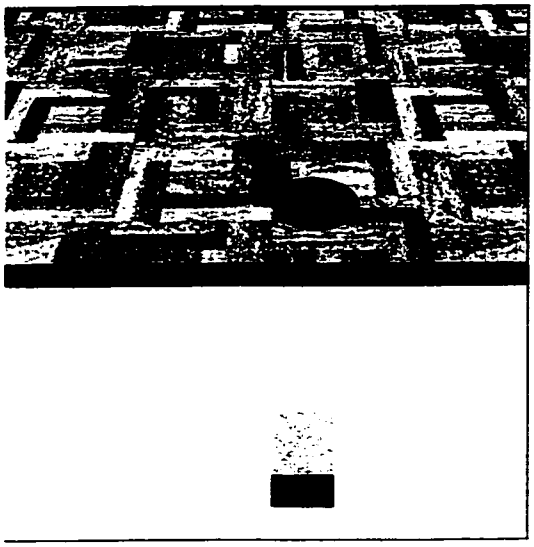
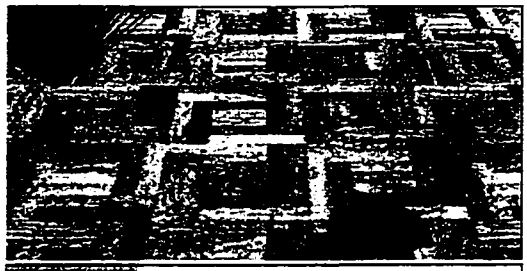
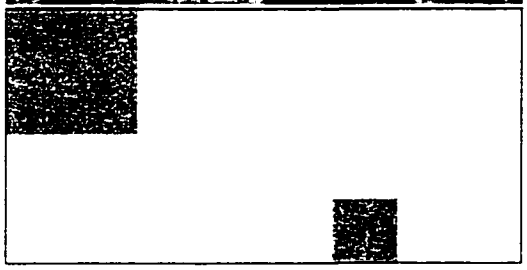


Fig. 4.11 A circular hole with a 10cm diameter is detected by the smallest fields of the array.

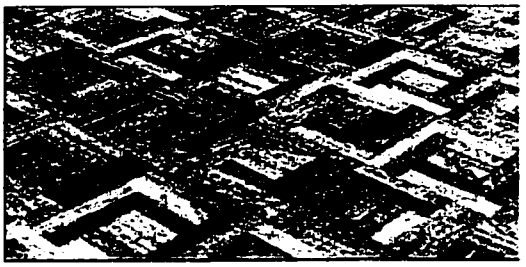
When multiple obstacles are in the field-of-view, such as in Fig. 4.12 (a), the foveated image shows that a correct steering decision is difficult to make. However, with a weighting scheme, such as that shown in Table 2, the closer object will automatically have a greater effect. Fig. 4.12 (b) and (c) show how the simulated robot correctly moves to the left first to avoid the circular hole which is closer to it.



(a) Frame1: bush obstacle appears at top left and hole obstacle appears at bottom right near center.



(b) Frame 2: robot correctly steers to the left to avoid the nearer obstacle first.



(c) Frame 3: obstacles are now out of view as robot steers to the left as seen by the pattern on the floor.

Fig. 4.12 A sequence of frames produced as two obstacles are encountered.

The results for each simulation run were taken from the top-view of the room which showed the path of the robot as it moved around the obstacles, as shown in Fig. 4.13. The size of the room was 5m on each side. The black line represents the center of the robot's path and the grey portion represents the width of the robot's body, which was set to 50cm. As mentioned, the two types of obstacles, a small green bush and a circular hole, were spread out across the floor.

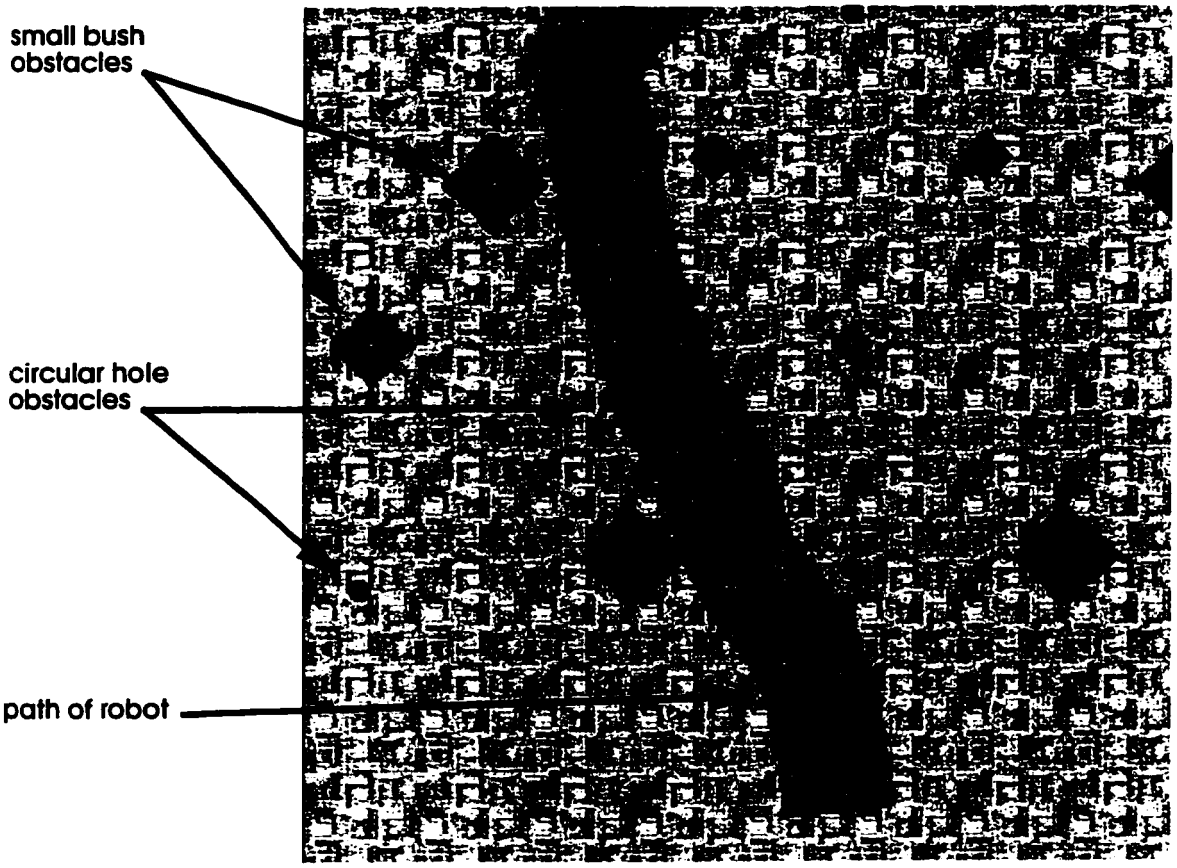
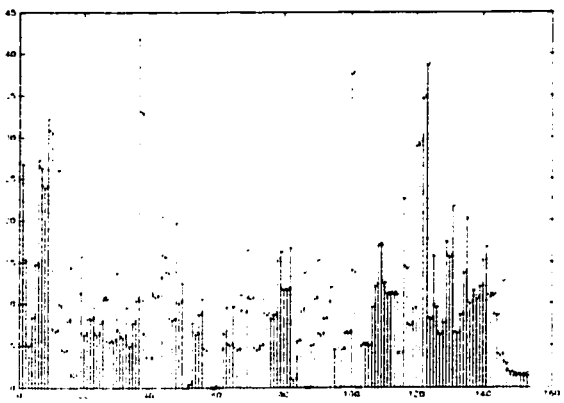
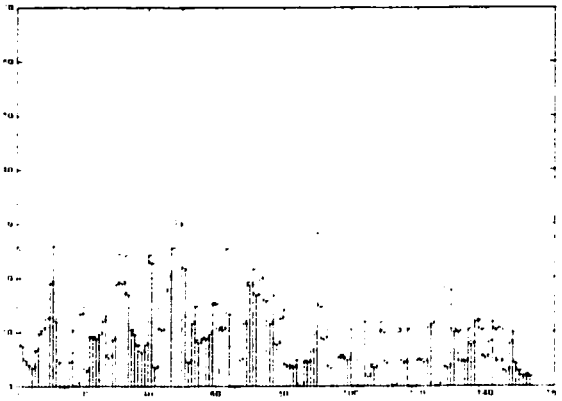


Fig. 4.13 Top-view of the room showing the obstacles and the path of the robot.

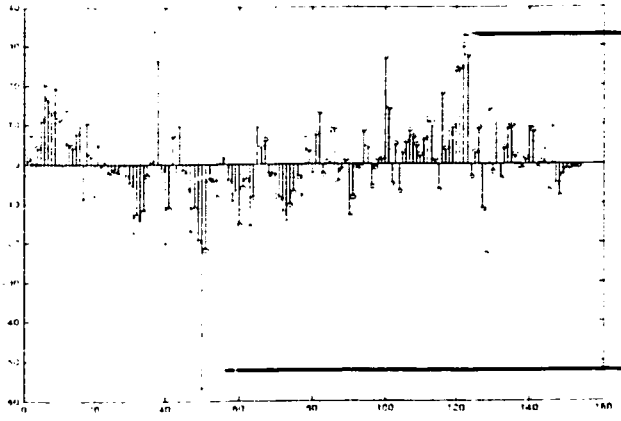
An example of a simulation run is shown in Fig. 4.14. The left control signal, $c_L(t)$, produced by the simulation, is shown in (a) and $c_R(t)$ is shown in (b). The difference between the two, shown in (c), was used to determine the orientation angle as in equation (15) of Section 4.3.2.2. To keep the angle from exceeding 8°, the parameter K_α, was set to a value between 1/5 and 1/10.



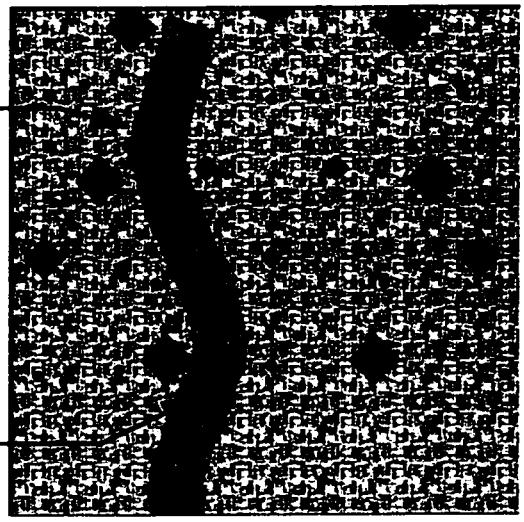
(a) Left control signal $c_L(t)$.



(b) Right control signal $c_R(t)$.



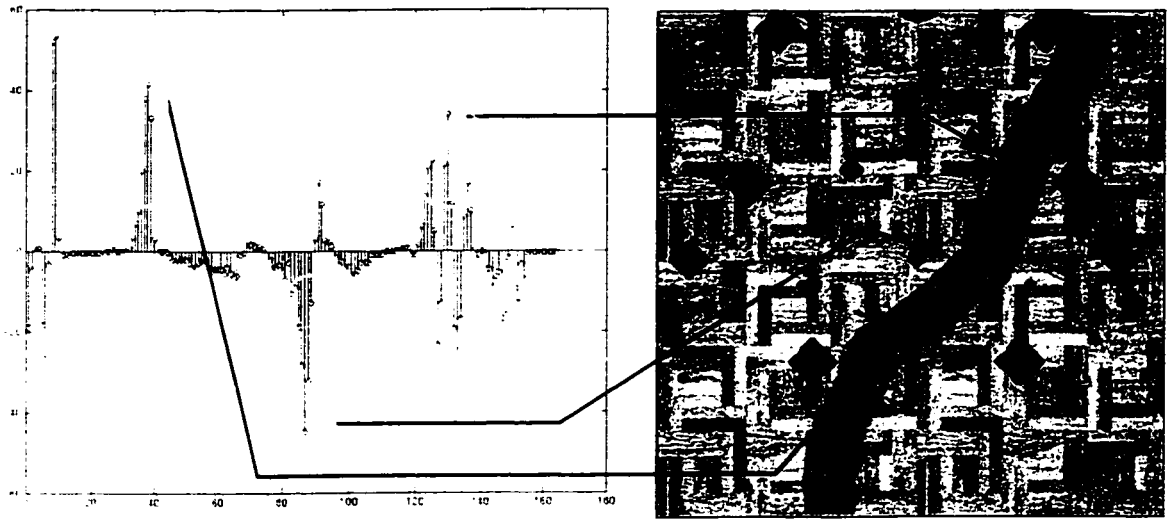
(c) Difference signal, $c_L(t) - c_R(t)$.



(d) Top-view of room showing the path

Fig. 4.14 Results for the left and right control signals and the steering signal for a particular run.

A similar example is shown in Fig. 4.15. This time the texture on the floor has been made more coarse by a factor of 5. The results show that this has no negative effect on performance. In fact the difference signal is easier to correlate with the path of the robot.



(a) Difference signal, $c_L(t) - c_R(t)$. (b) Top-view of room showing the path

Fig. 4.15 The steering signal for a simulation run using a coarse texture on the floor.

A set of four simulation runs are shown in Fig. 4.16. The first three runs, (a)-(c), were performed with a floor made to resemble a typical hardwood pattern. The fourth run, shown in (d), was performed on a floor of uniform color without texture. The runs were started from different positions so that obstacles would be encountered in different ways for each run. In each case the robot avoided the obstacles and generally avoided any contact between its body (in grey) and the obstacles.

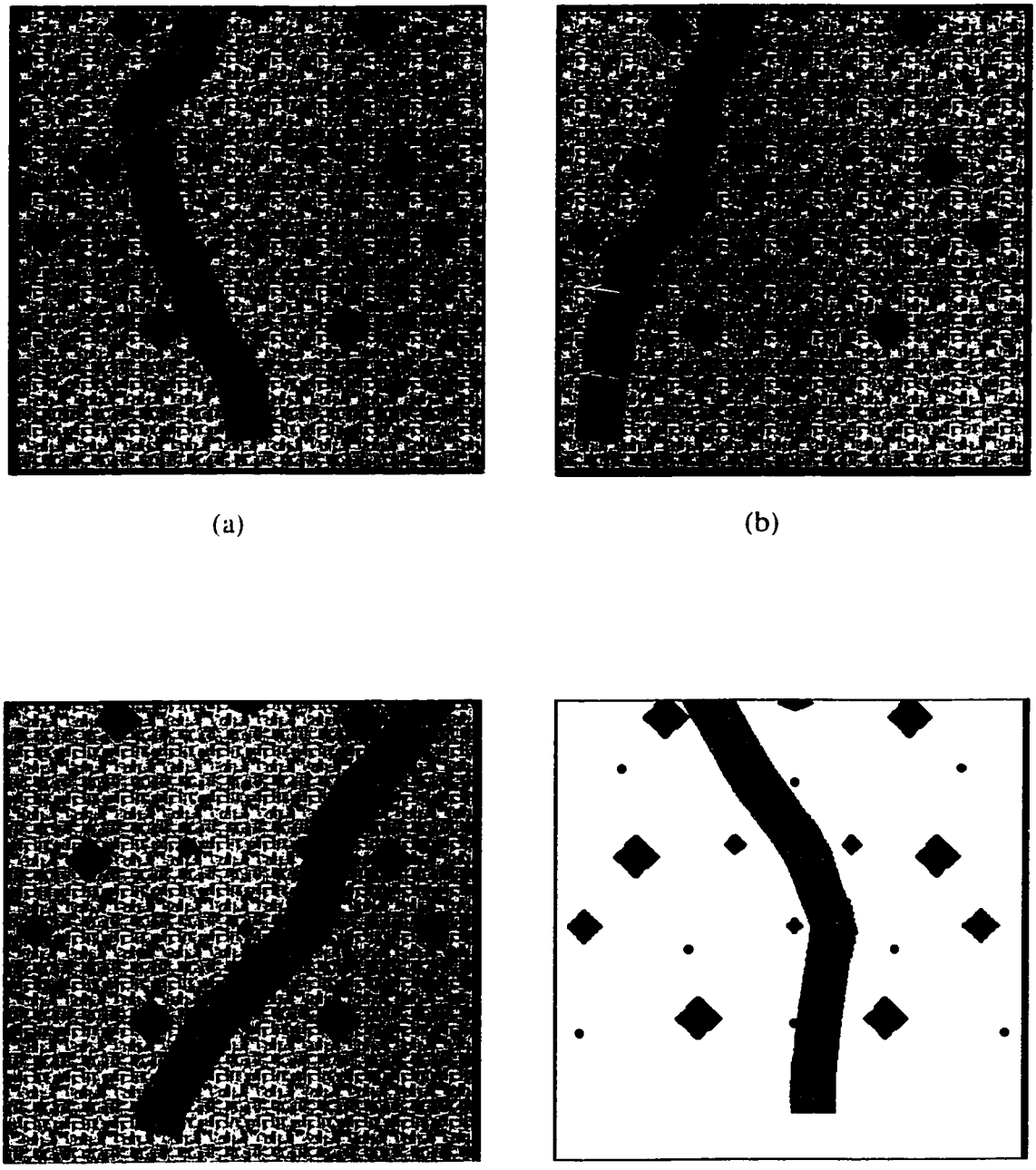
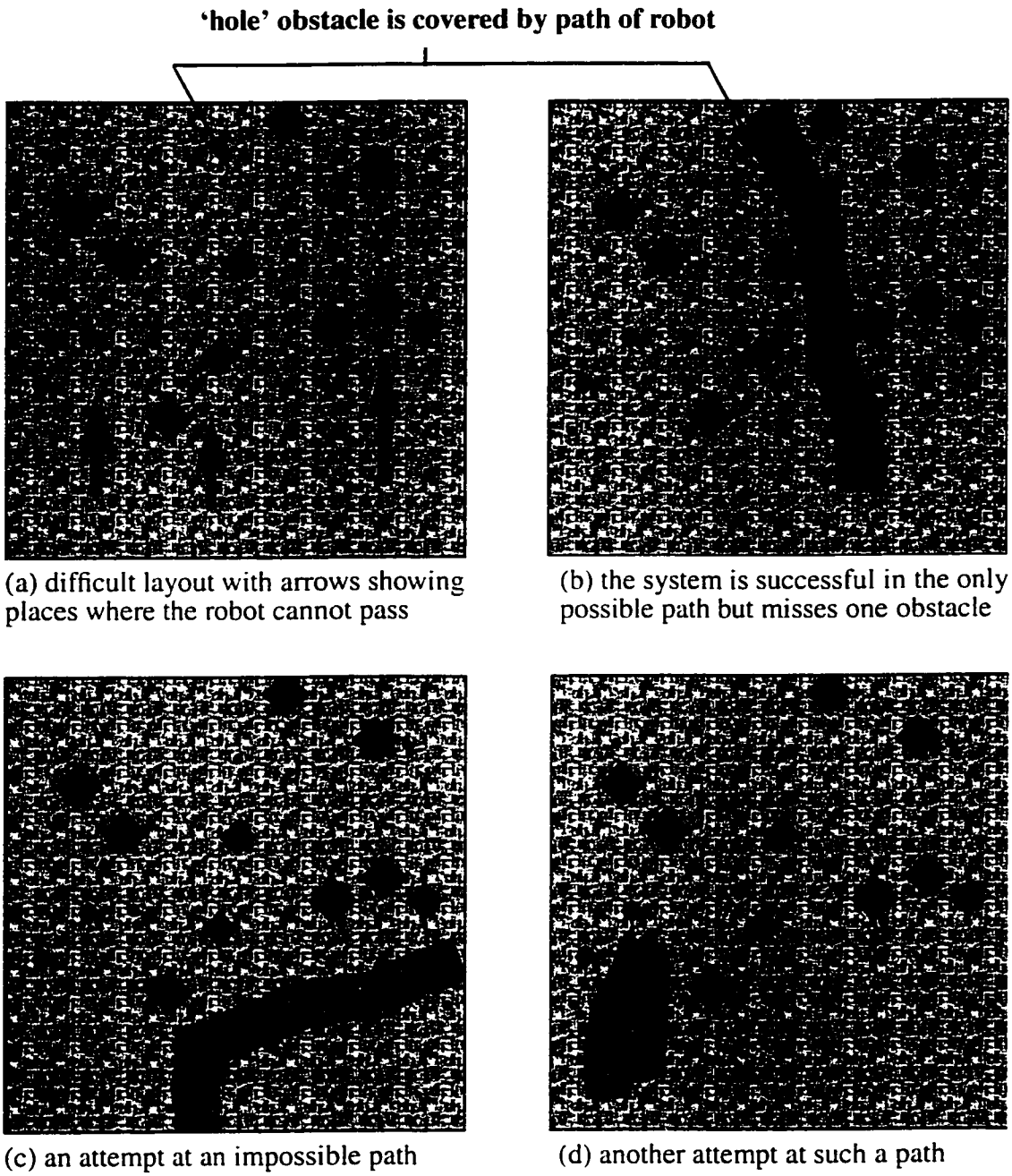


Fig. 4.16 Simulation results: (a)-(c) shows the results for a textured floor and (d) shows the results for an untextured floor.

Another configuration of obstacles was designed to create a more challenging scene to test the system. Fig. 4.17 (a) shows how the obstacles were arranged too closely for the robot to pass. Particularly difficult places are indicated with arrows. Fig. 4.17 (b) shows the only possible path for the robot. It was mostly successful, however, near the end of its course it passes over a 'hole' obstacle as indicated on the figure. The infraction is on the outer edge of the robot's body and is not surprising given how close the obstacles are. Fig. 4.17 (c) and (d) show how the system handled impossible situations. In both cases the system approached the obstacles until they became close enough to cause a sharp turn, due to the stronger weighting for obstacles in the fovea. The path taken in (d) was the result of two consecutive sharp turns and resulted in a total retreat from the area.



(a) difficult layout with arrows showing places where the robot cannot pass

(b) the system is successful in the only possible path but misses one obstacle

(c) an attempt at an impossible path

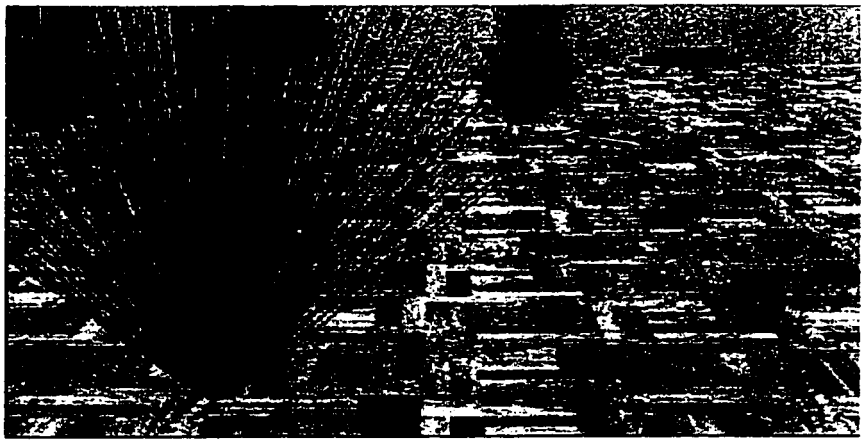
(d) another attempt at such a path

Fig. 4.17 Simulation results for a layout designed to make it difficult for the robot to pass by placing obstacles too close together.

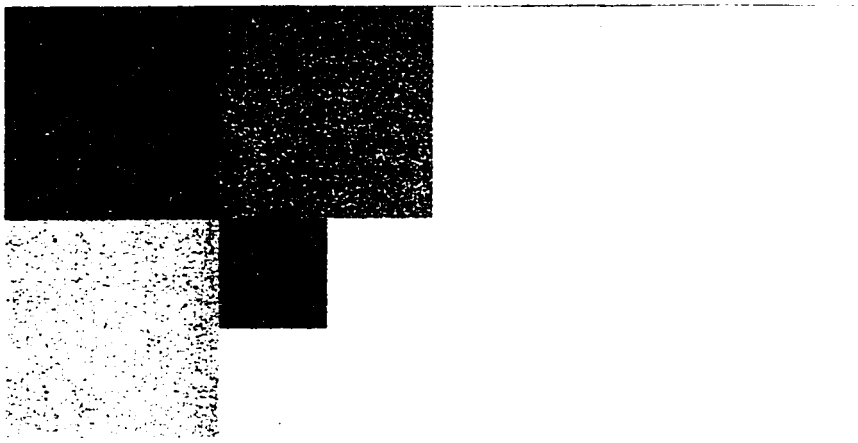
4.4 Conclusion

A simulation of the architecture described in Chapter 3 has been conducted with a still image and with a virtual environment. The results of the simulation with a virtual environment showed that a system in which the architecture is implemented is able to efficiently perform obstacle avoidance in an unstructured environment. In particular the results showed that the architecture was effective with or without textures in the scene which is an advantage over approaches based on stereo [36] or optical flow [34], which depend on the presence of surface details. They also showed that the weighting scheme combined with the post-receptor foveation makes it possible to appropriately avoid a combination of multiple obstacles depending on their place in the visual field and their position with respect to the body of the robot. The complex obstacle that looked like a bush helped to demonstrate the effectiveness of the architecture in avoiding obstacles which would be problematic for other approaches. Infrared sensing schemes which rely on reflected signals would not be able to detect the full extent of a bush-like object. Classical edge detection schemes would also have difficulty. Fig. 4.18 shows how a classical edge detector applied to the image of the bush results in a barrage of data which has parts of the bush intermingled with the pattern on the floor. The foveated image in (b) however, delivers a signal that can be easily used for a steering decision.

In general, the simulations showed that a smart sensor architecture could be used as an effective 2D solution to the 3D problem of obstacle avoidance by appropriately partitioning the processing in accordance with the task to be performed.



(a) Image of the 'bush' obstacle seen from the robot's point of view.



(b) Post-receptor foveation of the above image.

(c) Classical edge-detection applied to the image in (a).

Fig. 4.18 A comparison of images resulting from post-receptor foveation (b) and classical edge-detection (c), of an image of the 'bush' obstacle on a hardwood floor.

5

An Example Implementation of the Architecture Using a VLSI Circuit

This chapter presents an example of a circuit that could be used for the implementation of the sensory-motor architecture described in Chapter 3. The purpose is to demonstrate the feasibility of implementing the architecture using simple circuit building blocks with a standard CMOS technology.

5.1 From Architecture to Electronic Circuit

Smart sensor arrays need to be designed for standard CMOS technology so that the circuits can be easily integrated on the same chip as digital circuits. For a smart sensor array, the per-pixel processing circuitry must be very compact to have a reasonably high resolution for a given circuit area. For this reason the signal processing is usually implemented with analog circuits. The processing required by the architecture in this thesis is shown again in Fig. 5.1. The four essential operations are: averaging, differencing, scaling and summation.

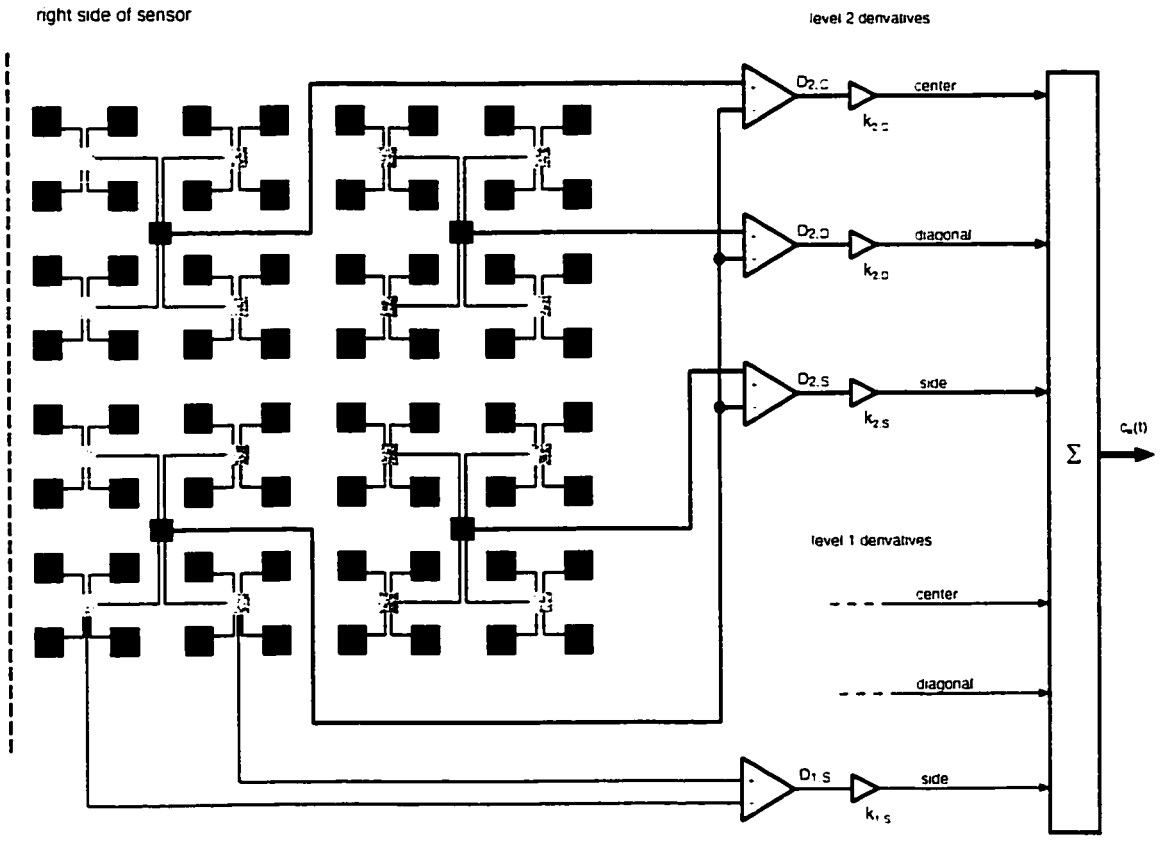


Fig. 5.1 Right side of the sensory-motor architecture as shown in Fig. 3.10 of Chapter 3.

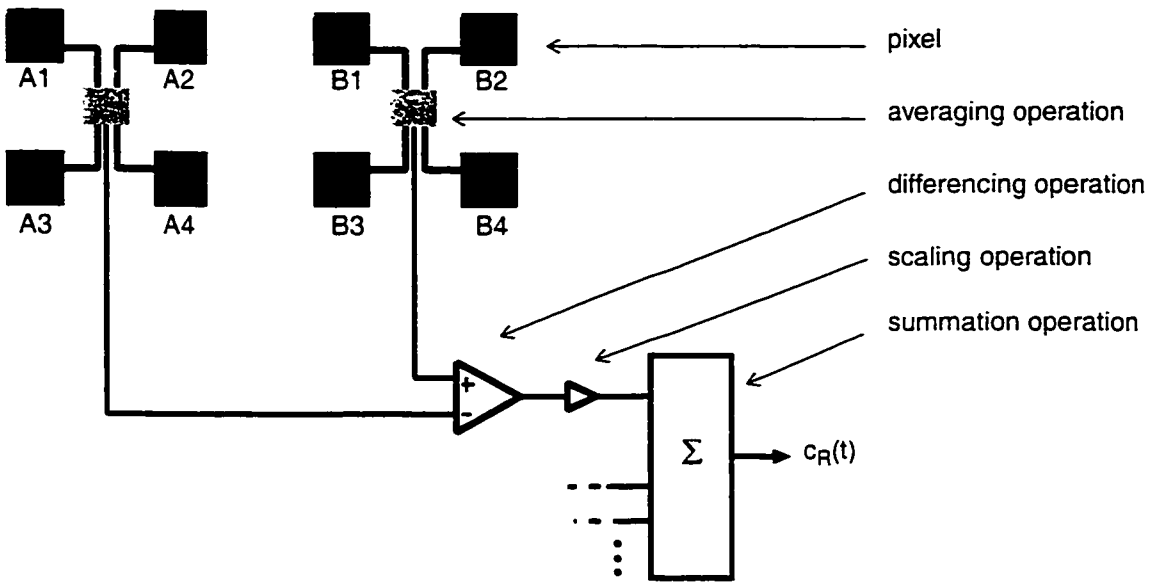


Fig. 5.2 A 4x2 portion of the architecture in Fig. 5.1 with the four required operations.

Fig. 5.2 shows a detail of the architecture with two fields of four photoreceptors each. It contains the four required operations. In CMOS technology, a photodiode serves as a photoreceptor. In an n-well process it can be formed from the junction between the p-substrate and the n++ diffusion, or that between the p++ diffusion and an n-well. As a result of the absorption of photons, each photodiode produces a photocurrent, the magnitude of which depends on the photonic power received, the size of the photodiode and the particular CMOS technology used.

An example of a possible current-mode circuit for averaging and differencing in a 4x2 portion of an array is shown in Fig. 5.3. The circuit makes use of Kirchoff's Current Law (KCL) to sum the incoming photocurrents. For example, the current I_A is the summation of the photocurrents I_{A1}, I_{A2}, I_{A3}, and I_{A4} into node A. The current I_A can be considered to be the average of the four photocurrents by consistently disregarding the division by four for each level. The same applies to current I_B at node B.

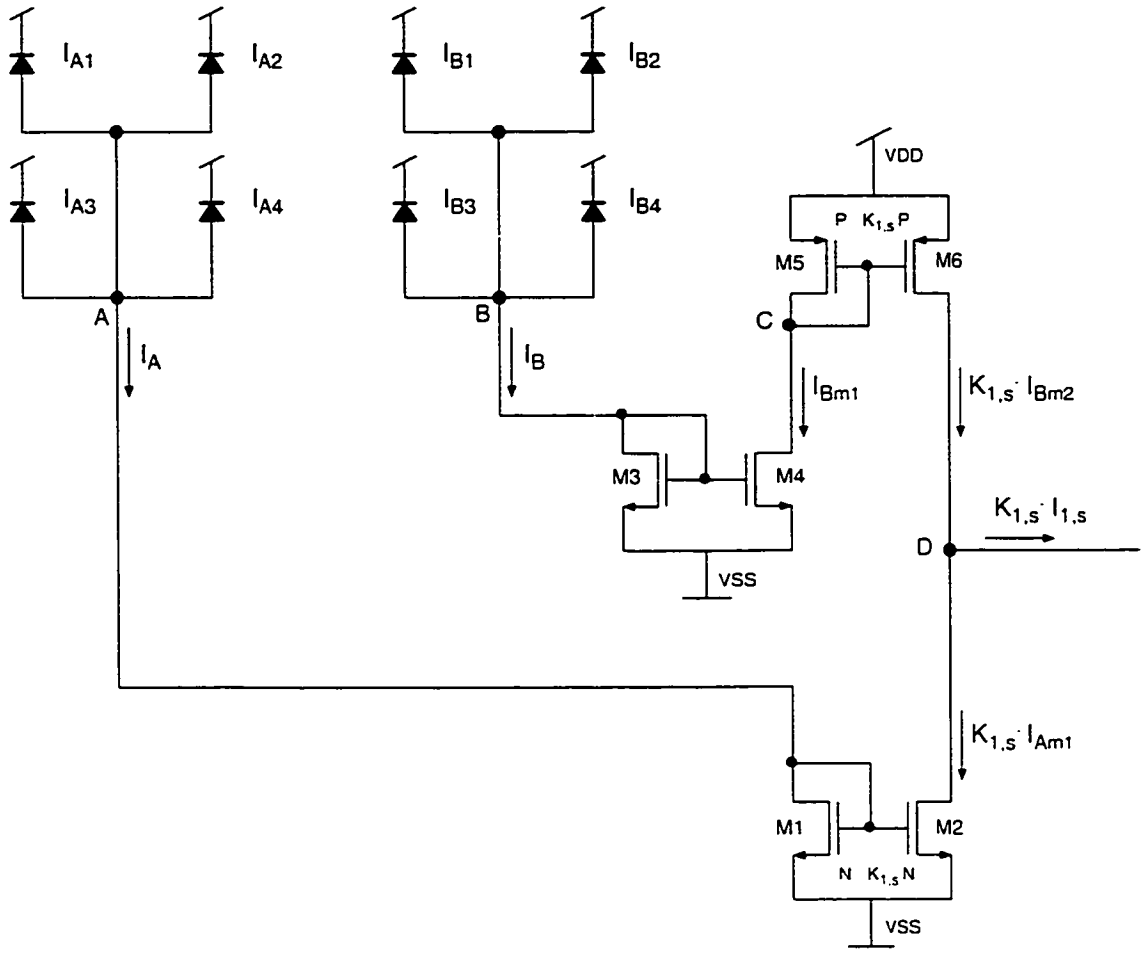


Fig. 5.3 Averaging, differencing and scaling circuits based on KCL.

The difference between the average currents I_A and I_B can be found by using KCL again. This time the direction of the currents are arranged at node D. by using current mirrors, such that the current I_{diff1} is equal to I_{Bm2} - I_{Am1}. The current mirror consisting of transistors M1:M2 is used to reverse the direction of I_A so that it is leaving node D as I_{Am1}. The current mirrors of M3:M4 and M5:M6 are used to reverse I_B twice in order to have node D between two mirrors. This makes it possible to control the voltage at node D. The

final two current mirrors (M1:M2 and M5:M6) can also be used for the scaling operation. Their output currents can be scaled by a factor $K_{1,S}$, as shown in Fig. 5.3, by adjusting the aspect ratios of the transistors appropriately. For example, the $K_{1,S}$ in the diagram would be:

$$K_{1,S} = \frac{(W/L)_{M2}}{(W/L)_{M1}} = \frac{(W/L)_{M6}}{(W/L)_{M5}}$$

Thus, the output current $I_{1,S}$ has a magnitude equal to the difference of I_A and I_B, scaled by a factor $K_{1,S}$. The direction of the current is determined by the sign of the difference. For the processing, only the magnitude of the difference is important and the current should always be positive. Therefore, an absolute value operation should follow the differencing. The circuit in Fig. 5.4 can be used to produce the absolute value of the difference current, $I_{1,S}$ [37]. The operation of the circuit can be understood by seeing that no current will pass through transistor M7 or M8 as long as the voltage at node D is close to $V_{DD}/2$. If $I_{Bm2} > I_{Am1}$ then the voltage at node D will rise. At some point the voltage will be large enough for M7 to be turned on and M8 to be off and the difference current will pass through it. If $I_{Bm2} < I_{Am1}$ then the voltage at node D will drop until it is low enough for M8 to conduct and M7 will be off. In both cases, the current through M7 or M8 will be redirected through one of the current mirrors to produce a unidirectional output current.

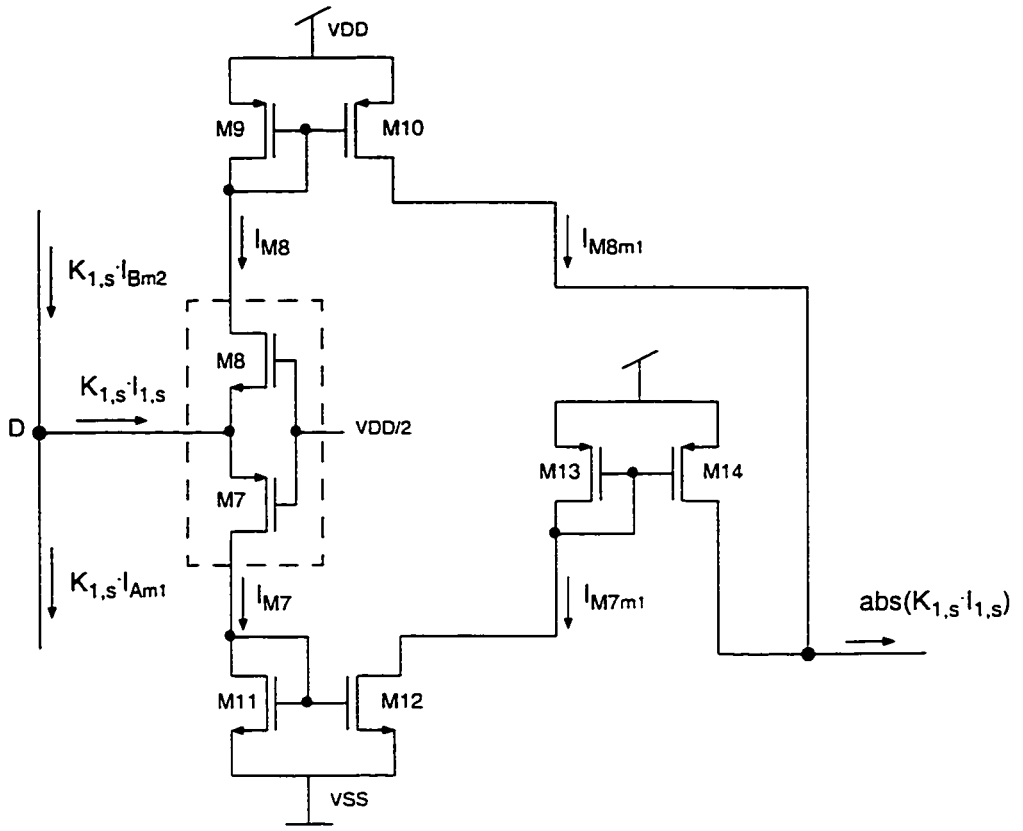


Fig. 5.4 An absolute value circuit to follow the circuit in Fig. 5.3.

The circuits in Fig. 5.3 and 5.4 were simulated using models for a 0.18μ CMOS process with V_{DD} = 1.8V. The current mirrors were all configured as cascode mirrors to increase their output resistance. The results of the simulation are shown in Fig. 5.5. The photocurrent I_B (consisting of the sum of four photocurrents) was held constant at 400nA while I_A varied from 300nA to 500nA by varying one of its photocurrents from 0 to 200nA. The output current shown at the top of the figure always reflects the magnitude of the difference between I_A and I_B. The voltage at node D is also shown at the bottom.

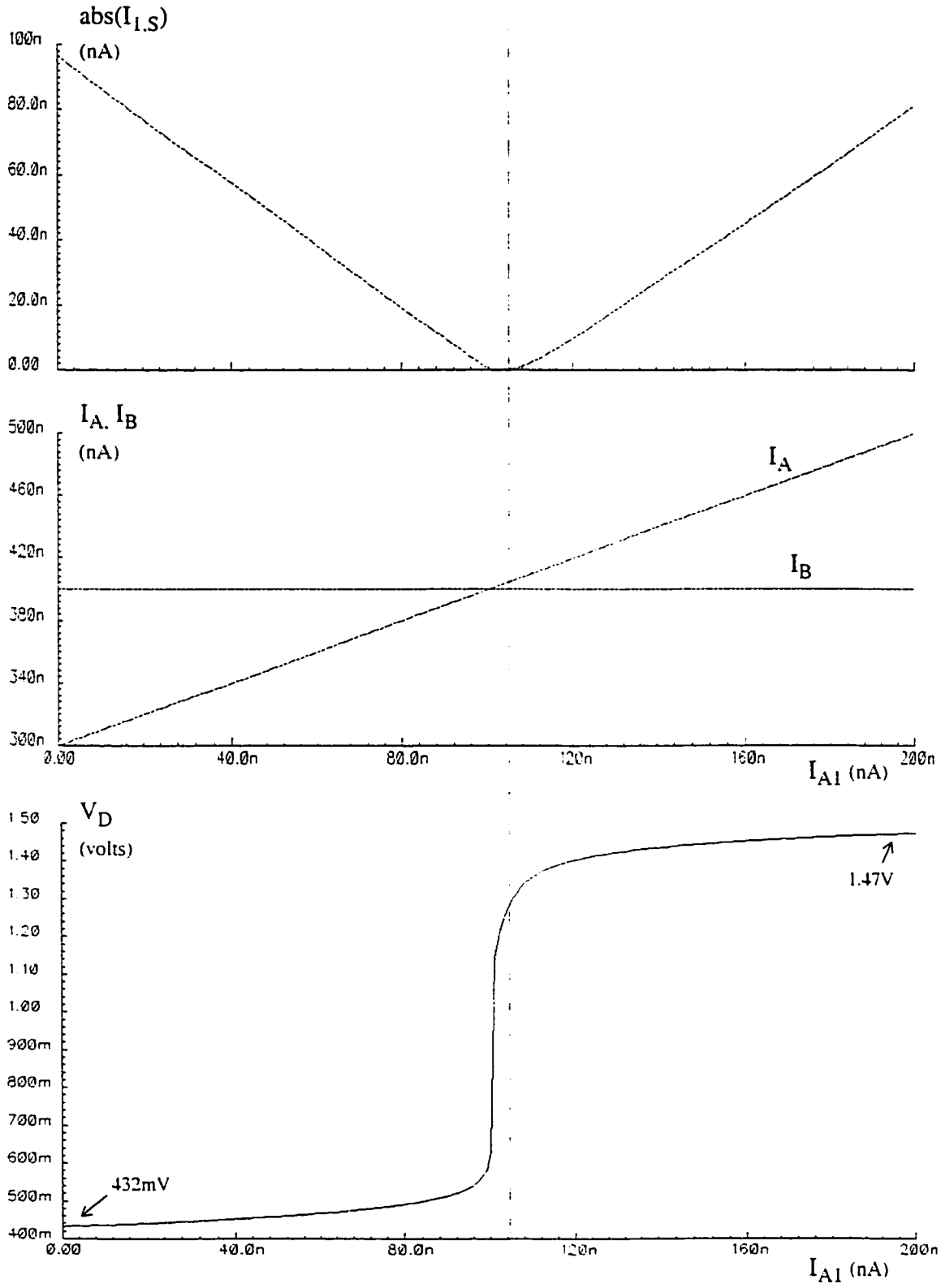


Fig. 5.5 The output current (top) with all photocurrents at 100nA except I_{A1} which goes from 0 to 200nA. The output current reflects the difference between I_A and I_B.

5.2 Output Stage of the Sensor Array

The current-mode circuits presented in the previous section are convenient for operations like addition and subtraction and occupy a small area. However, it is easier to output a voltage signal than a current signal. A current signal with varying magnitude is very susceptible to noise. To facilitate the signal output, the magnitude varying current signal is converted into a stream of pulses with constant amplitude and width, but the duration of the pulse cycle is proportional to the magnitude of the current, as shown in Fig. 5.6 [38].

A circuit unit that can be used to convert a current signal to a cycle-time varying pulse signal is shown in Fig. 5.7 [40]. It consists of a schmitt trigger and an inverter with a feedback path that makes the capacitance (C_s) to be charged and discharged alternatively. This forms an 'integrate and fire' structure in which the capacitor, C_s, can be charged up to the upper threshold of the schmitt trigger by the input current. The resulting 'high' output will cut-off the charging switch, M17, to the capacitor and turn-on the discharge switch, M16, to lower the voltage on the capacitor to the lower threshold of the trigger, sending the output to 'low'. The voltage, V_b, on the transistor M15, below the discharge switch, is used to control the discharge current, thus controlling the pulse width, whereas the input

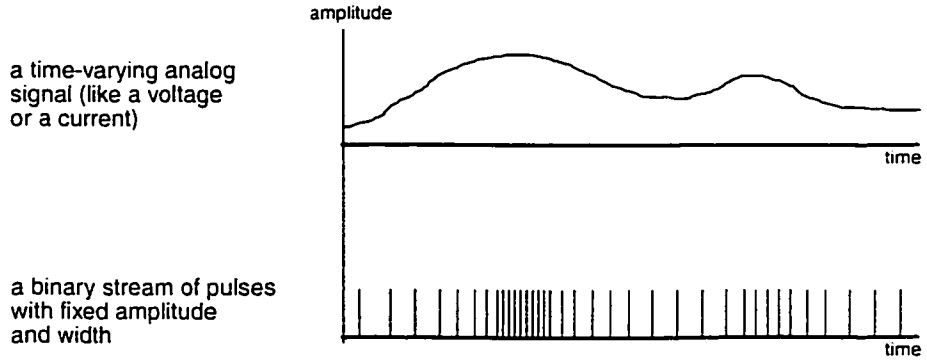


Fig. 5.6 Converting a magnitude-varying signal to a frequency-varying one.

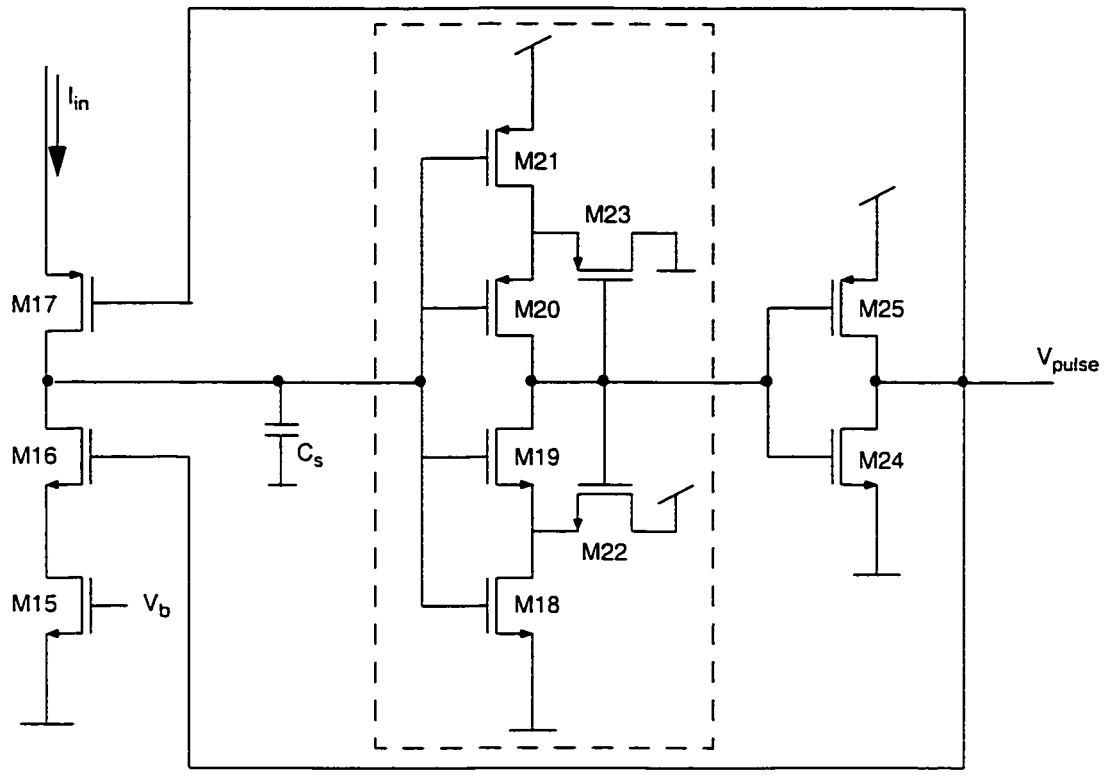
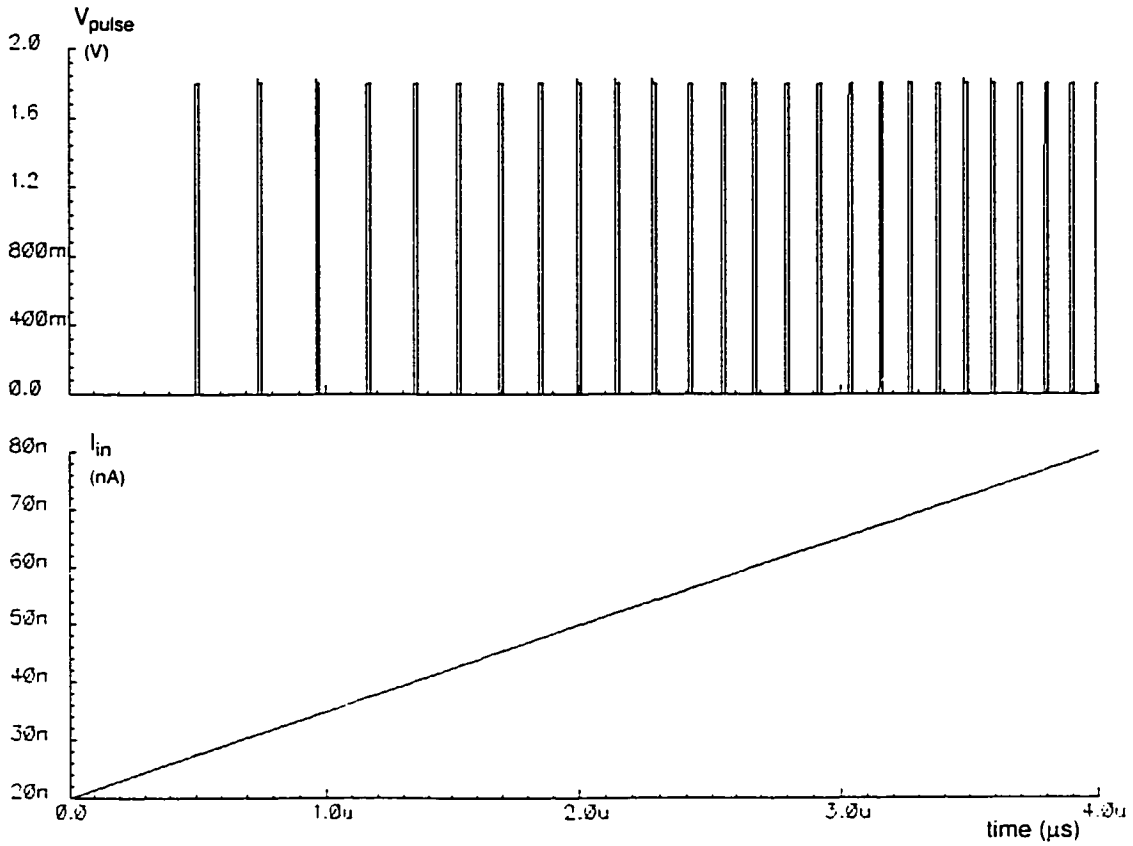


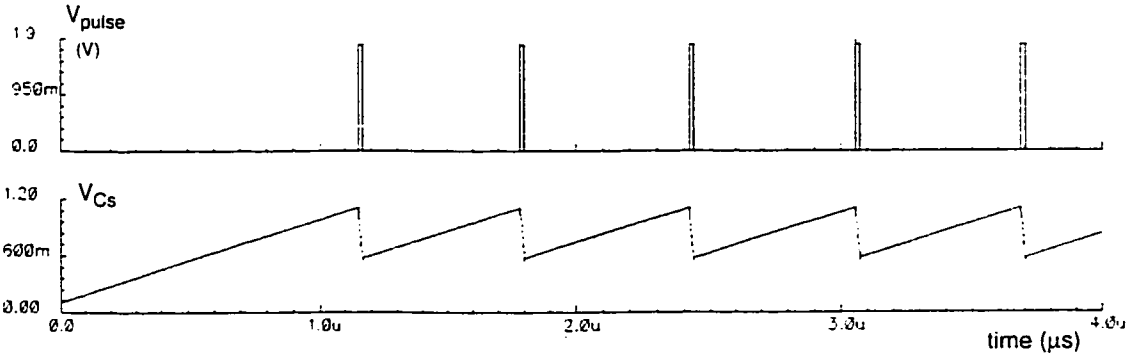
Fig. 5.7 Basic circuit to convert a current signal to a pulse stream of a particular frequency. The dashed box encloses the schmitt trigger [40].

current determines the pulse frequency. The schmitt trigger provides for a clean transition between voltage levels and the hysteresis helps to avoid unstable conditions.

The results of a simulation are shown in Fig. 5.8. The output pulse train for an input current ranging from 20nA to 80nA is shown in (a). The time between pulses decreases gradually as the current increases. In (b) an output pulse train is shown for an input current of 20nA. The voltage at the capacitor, V_{Cs}, is shown below it to indicate how the charge and discharge pattern of the capacitor leads to each output pulse. In both cases the pulse width is approximately 10ns.



(a) Pulse train output for an input current ranging from 20 nA to 80nA.



(b) Pulse train output for an input current of 20 nA. The bottom waveform shows the voltage on the capacitor, C_s, as it charges and discharges.

Fig. 5.8 Simulation results for the 'integrate and fire' circuit of Fig. 5.7.

5.3 Conclusion

A set of circuit units have been presented to provide a possible implementation example for the architecture described in Chapter 3. It has been shown that the required operations can be implemented easily with a simple current-mode circuit, which confirms that the resulting pixel size will be very small, and the sensor circuit can be very easily implemented using a standard CMOS technology.

6

Conclusion

The work of this thesis was directed at solving the problems of existing sensory-motor systems for obstacle avoidance in autonomous robots by developing a new architecture targeted to a smart sensor implementation. The problems of existing systems which were specifically addressed were the large volume of data to be processed, the computational resources required for the processing, and the hardware resources required for the

implementation.

Extracting relevant information about real-world obstacles is a difficult task. There have been a variety of VLSI sensory-motor systems developed in the past decade intended to replace traditional microprocessor-based computer vision systems. However, because the processing in these systems is at the pixel level, it is difficult to achieve algorithms that can deal with real-world, unstructured environments. It was concluded that new strategies and architectures would be required to get past these limitations.

In this thesis, a new sensory-motor architecture has been proposed. Its development was based on inspiration drawn from behavior-based robotics and the biology of natural vision systems. Emphasis was placed on the particular needs of an obstacle avoidance behavior for a simple wheeled robot. Using this approach led to the two organizing principles of the architecture: (a) a special post-receptor foveation scheme designed to emphasize parts of the visual signal which are relevant to obstacle avoidance, and (b) the alignment of sensory and motor maps to more easily convert the visual signal to a control signal to be used to steer around obstacles.

In the post-receptor foveation scheme, hierarchical averaging is performed on a homogenous array of photoreceptors. Unlike with other foveation schemes that vary the size and arrangement of photoreceptors, the pixels can be more like standard cells with a constant pitch, designed to abut with common power rails and interconnections across the array. The use of hierarchical averaging makes all levels of resolution available for processing, supporting spatiotemporal operations down to the pixel level. From the point of view of obstacle detection, post-receptor foveation simplifies obstacle detection by producing a type of low-pass filtering which makes complex objects (like bushes or rocks)

appear like a block of uniform intensity while retaining enough information for obstacle avoidance. This can greatly reduce the amount of computation necessary to localize an obstacle relative to the fovea.

In the alignment of sensory and motor maps, the center of the sensor is aligned with the midline of the robot's body. This means that the point of fixation and the visual field are also aligned with the body. Whether an obstacle lies more on the left or right side of the sensor can be quickly determined using spatial derivatives and other visual cues, and can lead to a quick decision to steer to the right or the left. Just as in sensory-motor parts of the brain, a stimulus in a sensory map causes activity in a motor map permitting rapid orientation behaviors. The focal-plane processing of a smart sensor is a good substrate for this type of scheme because the map of the visual field is always present across its photo-receptor array. The scheme was further enhanced by associating regions of the visual field with the body of the robot such that visual cues could be weighted according to their relative proximity to the robot. With this refinement, objects which are more immediate and closer to the center will elicit a stronger signal for the robot to turn away from them. Thus, a sufficiently accurate sensory-motor transformation is achieved without the use of explicit coordinates or time-consuming optical flow computations

A detailed model of the architecture and the imaging process was described in Chapter 4. The model was used to develop an algorithm for a simulation of the architecture. The algorithm highlighted the advantages of using a parallel approach to performing imaging operations because of their simple and iterative computations and because of the large volume of data. A simulation scheme, which takes into account a realistic environment, has been developed to show that the sensory-motor architecture is able to perform obstacle

avoidance in an unstructured environment. Three-dimensional computer graphics were used to create obstacles which have complex shapes and irregular contours. The example of the architecture used in the simulation is a sensor array with a relatively low resolution of 16x8 pixels. The effectiveness of the approach was demonstrated by simulating the passage of a robot through a group of obstacles on both textured and untextured floor surfaces. The weighting scheme of the architecture was also shown to be effective in dealing with combinations of obstacles which are all in the field-of-view but which are in different locations with respect to the robot.

For the VLSI implementation of the architecture, circuit units to be integrated in each pixel for the signal processing have been proposed. The architecture makes use of only four signal processing operations: averaging, differencing, scaling and summation. Very simple current-mode circuits can be used to perform these operations. Integration of these circuit units do not require any special fabrication process. Moreover, the use of simple processing circuitry means that most of the area in each pixel can be used for the photoreceptor, resulting in a high fill-factor. This means that large photoreceptors can be used, increasing the signal level of the photocurrent and possibly reducing the variance of receptor response across the sensor.

To conclude, the significance of the sensory-motor architecture in this thesis resides in its ability to achieve a difficult task while using minimal computational and hardware resources. It enables the processing of a dense flow of visual information while using a relatively low resolution pixel array. The architecture leads to an effective 2D solution to the 3D problem of obstacle avoidance by appropriately partitioning the processing in accordance with the task to be performed.

Based on the work of this thesis, future work will involve an ASIC implementation of the architecture. Also of interest is an exploration of the inherent flexibility of the architecture with respect to the size and placement of the fovea. Different configurations may lead to improved performance for particular tasks and environments. Other applications for the architecture may also be profitably explored. The most promising are those which are related to obstacle avoidance, such as target acquisition and target tracking. The foveated structure and weighting scheme can possibly be used to keep an object centered on the fovea and thus track its changing position.

References

[1] L. Matthies, E. Gat, R. Harrison, B. Wilcox, R. Volpe, and T. Litwin, "Mars microrover navigation: performance evaluation and enhancement," *Autonomous Robots*, no. 2, pp. 291-311, 1995.

[2] M.A. Schuster and G. Strull, "A monolithic mosaic of photon sensors for solid state imaging applications," *IEEE Transactions on Electron Devices*, vol. ED-13, pp. 907-912, 1966.

[3] R. Dyck and G. Weckler, "Integrated arrays of silicon photodetectors for image sensing," *IEEE Transactions on Electron Devices*, vol. ED-15, pp. 196-201, April 1968.

[4] P. Noble, "Self-scanned silicon image detector arrays," *IEEE Transactions on Electron Devices*, vol. ED-15, pp. 202-209, April 1968.

[5] W.S. Boyle and G.E. Smith, "Charge-coupled semiconductor devices," *Bell Systems Technical Journal*, vol. 49, pp. 587-593, 1970.

[6] R.F. Lyon, "The optical mouse, and an architectural methodology for smart digital sensors," in *Proceedings of the CMU Conference on VLSI Structures and Computations*, Computer Science Press, October 1981.

[7] C. Mead, *Analog VLSI and Neural Systems*. New York: Addison-Wesley, 1989.

[8] M. Mahowald and C. Mead, "The Silicon Retina," *Scientific American*, vol. 264, no. 5, pp. 76-82, 1991.

[9] R. Wodnicki, G. Roberts, and M. Levine, "A foveated image sensor in standard CMOS technology," in *Proceedings of the Custom Intergrated Circuits Conference*, pp. 357-360, 1995.

[10] E.L. Schwartz, D.N. Greve, G. Bonmassar : "Space-variant active vision: definition, overview and examples", *Neural Networks*, 8, pp. 1297-1308, 1995.

[11] J.J. Clark and D.J. Friedman, "VLSI sensorimotor systems," in *Proceedings of the 1991 IEEE Robotics and Automation Conference*, Sacramento, pp. 1342-1347, April 1991.

[12] B.E. Stein and M.A. Meredith, *The Merging of the Senses*, Cambridge, MA: The MIT Press, 1993.

[13] R. A. Andersen, V.B. Mountcastle, The influence of the angle of gaze upon the excitability of light-sensitive neurons of the posterior parietal cortex. Journal of Neuroscience, pp. 532-548, 1983.

[14] J. Lazzaro, S. Ryckebusch, M.A. Mahowald, and C. Mead, "Winner-take-all networks of O(n) complexity," in *Advances in Neural Information Processing Systems 1*, Touretzky, Ed., San Mateo, CA:Morgan Kaufmann, 1988.

[15] M. Maris and M. Mahowald, "Neuromorphic sensory-motor mobile robot controller with attention mechanism", in *Neuromorphic Systems: Engineering Silicon from Neurobiology*, L. S. Smith & A. Hamilton, Eds., World Scientific, 1998.

[16] G. Indiveri, "Neuromorphic analog VLSI sensor for visual tracking: circuits and application examples", *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing*, vol. 46, no. 11, pp. 1337--1347, November 1999.

[17] R.R. Harrison and C. Koch, "A neuromorphic visual motion sensor for real-world robots", presented at the Workshop on Defining the Future of Biomorphic Robotics, IROS, Victoria, B.C., Canada, 1998.

[18] R. Etienne-Cummings, J. Van der Spiegel, P. Mueller and M.Z. Zhang, "A foveated silicon retina for two-dimensional tracking", *IEEE Transactions on Circuits and Systems II*, vol. 47, pp. 504-517, June 2000.

[19] L.M. Lorigo, R.A. Brooks and W.E.L. Grimson, "Visually-guided obstacle avoidance in unstructured environments," in Proceedings of IROS 1997, Grenoble, France, September 1997.

[20] I. Horswill, Specialization of Perceptual Processes. Ph. D. Thesis, Massachusetts Institute of Technology, May 1993.

[21] R.A. Brooks, "New approaches to robotics," *Science*, 253, pp. 1227-1232. 1991.

[22] R. C. Arkin, *Behavior-Based Robotics*. Cambridge, MA: MIT Press, 1998.

[23] G. Sandini and V. Tagliasco, "An anthropomorphic retina-like structure for scene analysis," *CGIP*, vol.14:365--372, 1980.

[24] J. van der Spiegel, G. Kreider, C. Claeys, I. Debusschere, G. Sandini, P. Dario, F. Fantini, P. Belluti & G. Soncini, "A foveated retina-like sensor using CCD technology," . In C. Mead & M. Ismail, editor, *Analog VLSI implementation of neural systems*, chapter 8, pp.189-212. Kluwer Academic Publishers, Boston, 1989.

[25] R. Wodnicki, G. Roberts, and M. Levine, "A foveatd image sensor in standard CMOS technology," in Proc. Custom Intergrated Circuits Conf., pp. 357-360. 1995.

[26] F. Pardo, J.A. Boluda, B. Dierickx, and D. Scheffer, "Design issues on CMOS spacevariant image sensors," in European Symposium on Advanced Imaging and Network Technologies, AFPAEC'96, Berlin, Germany, October 1996.

[27] S. Tanimoto and T. Pavlidis, "A hierarchical data structure for picture processing," Computer Graphics and Image Processing, 4: 104-119, 1975.

[28] A.B. Butler and W. Hodos, Comparative Vertebrate Neuroanatomy: Evolution and Adaptation. New York: Wiley-Liss, 1996.

[29] A. Duchon, "Robot navigation from a gibsonian viewpoint", IEEE International Conference on Systems, Man and Cybernetics, San Antonio, Texas, IEEE, Piscataway, NJ, pp 2272-2277, October 2-5, 1994..

[30] P. Sobey and M.V. Srinivasan, "Measurement of optical flow using a generalized gradient scheme," *J. Opt. Soc. Am.* 8, 1488-1498.

[31] C.M. Higgins, and C. Koch, "A modular multi-chip neuromorphic architecture for real-time visual motion processing," *Analog Integrated Circuits and Signal Processing* 24, 195-211, 2000.

[32] D. Ballard, "Animate vision", *Artificial Intelligence*, vol. 48, no. 1, pp. 1-27, February 1991.

[33] O. Faugeras, *Three-Dimensional Computer Vision*, MIT Press, November 19, 1993.

[34] D. Coombs, M. Herman, T. Hong, and M. Nashman, "Real-time obstacle avoidance using central flow divergence and peripheral flow," *IEEE Transactions on Robotics and Automation*, vol. 14, no. 1, pp. 49--59, 1998.

[35] Maxon Motor Catalogue, pp. 34-41, 2002.

[36] D. Murray, C. Jennings, "Stereo vision based mapping and navigation for mobile robots," *IEEE International Conference on Robotics and Automation*, Albuquerque, NM, 20-25, April 1997.

[37] Z. Wang, "Current-Mode CMOS Integrated Circuits for Analog Computation and Signal Processing," *Analog Integrated Circuits and Signal Processing*, 1, pp. 287-295, 1991.

[38] A.F. Murray and A.V.W. Smith, "Asynchronous Arithmetic for VLSI Neural Systems," *Electronics Letters*, vol. 23, no. 12, pp. 642-643, June, 1987.

[39] J.E. Tomberg, and K. Kaski, "Pulse-density Modulation Technique in VLSI Implementations of Neural Network Algorithms", *IEEE J. Solid State Circuits*, Vol. 25, pp. 1277-1286, Oct. 1990.

[40] J. Meador, A. Wu, C. Cole, N. Nintunze and P. Chintrakulchai, "Programmable Impulse Neural Circuits", *IEEE Transactions on Neural Networks*, Vol. 2, No.1, pp. 101-108, January 1991.

Appendix A Derivation of u_0 and v_0

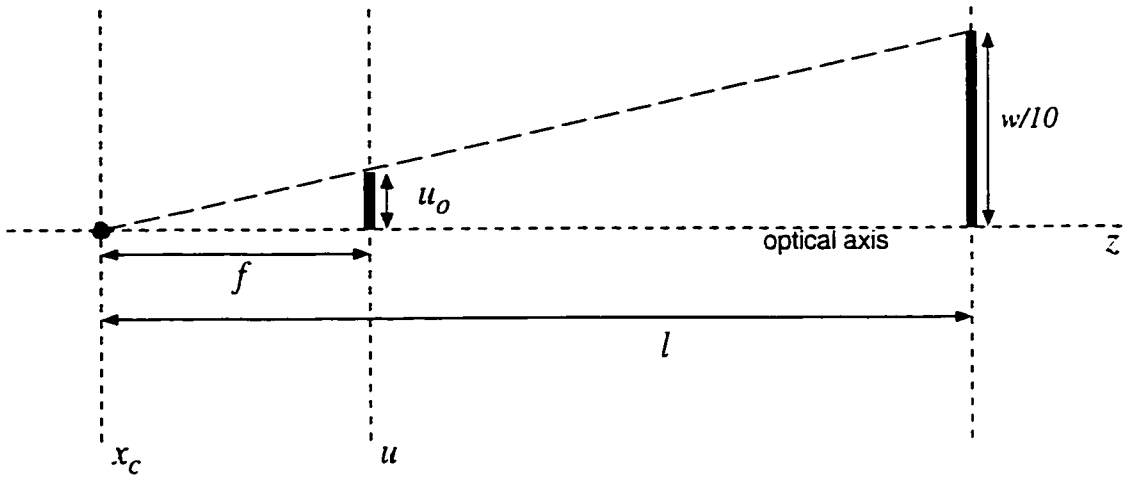


Fig. A.1 Top view of the imaging geometry showing the similar triangles of perspective projection used to determine the image width of the smallest feature.

The ratios from the similar triangles in Fig. A.1 are:

$$\frac{w/10}{l} = \frac{u_o}{f} \tag{A.1}$$

and solving for u_0 gives:

$$u_o = f \cdot \frac{w/10}{l}. \tag{A.2}$$

The length from the center of projection to the point of fixation is given by:

$$l = \frac{h}{\cos(\theta)} \tag{A.3}$$

which leads to an expression for u_0 in terms of the vertical gaze angle θ:

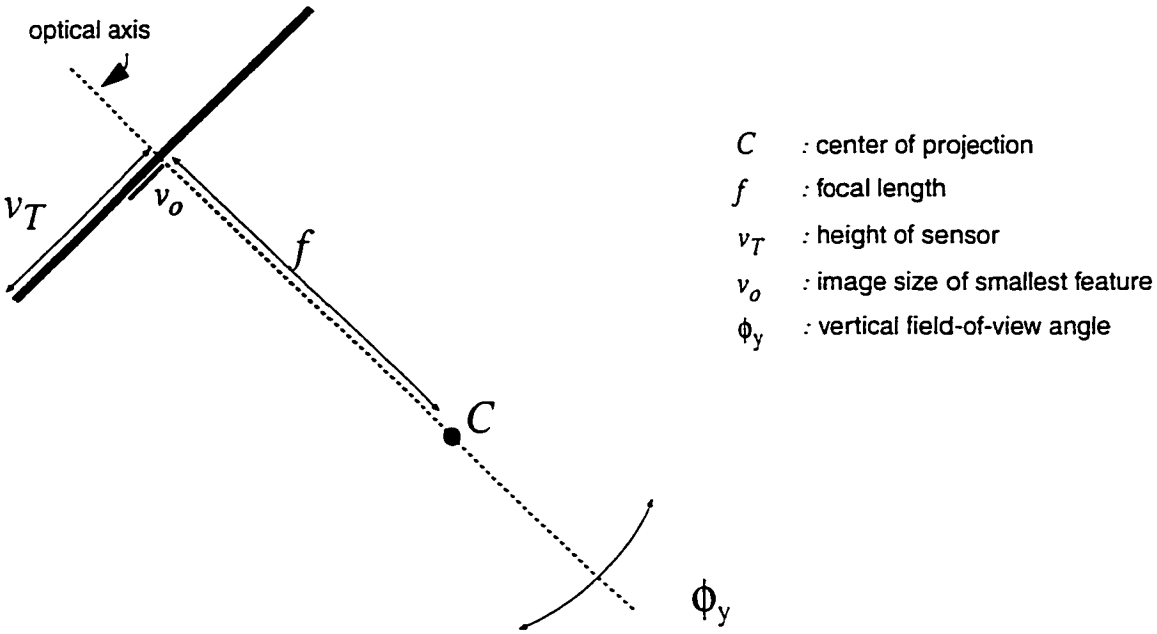
$$u_o = f \cdot \frac{w}{10h} \cdot \cos(\theta). \tag{A.4}$$

If the width of the robot and height of the sensor mounting and its focal length are considered constants, they can be combined as:

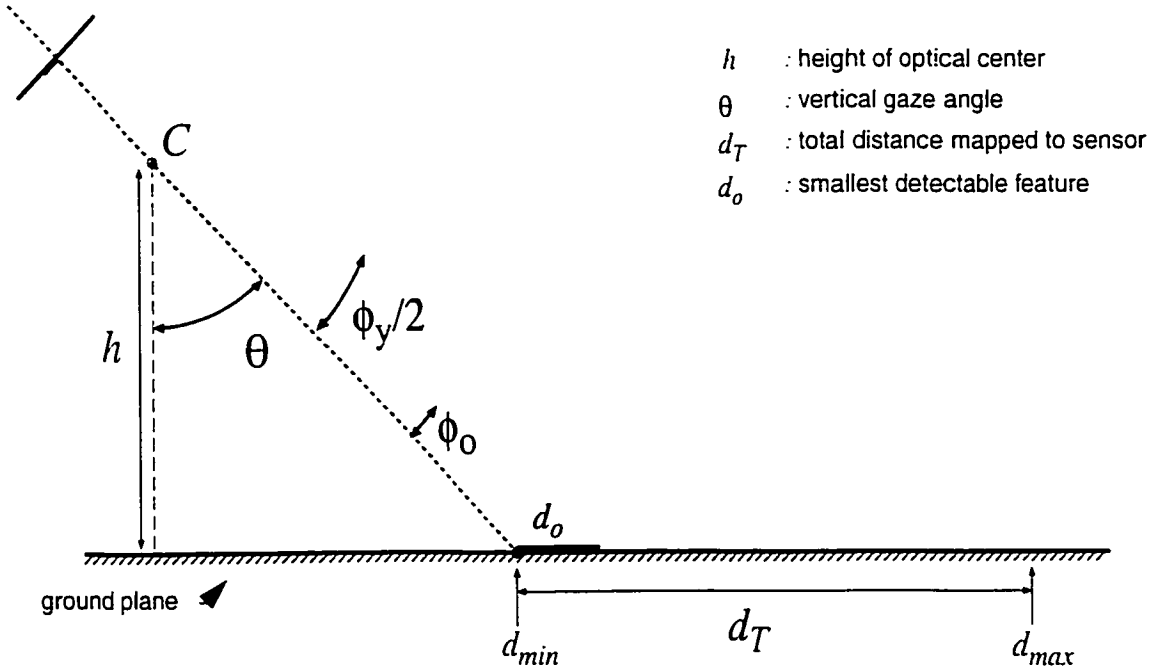
$$K_b = f \cdot \frac{w}{10h} \tag{A.5}$$

and then u_0 can be expressed as a function of the vertical gaze angle:

$$u_o = K_b \cdot \cos(\theta) \quad . \tag{A.6}$$



(a) Focal length should be selected to produce an image for the size of the sensor.



(b) The height and angle of the sensor with respect to the ground plane determine the mapping of features on the ground plane to the surface of the sensor.

Fig. A.2 Imaging geometry used to determine the size of the smallest feature to be detected.

From Fig. A.2 it can be seen that the intersection of the field of view of the sensor with the ground plane is bounded by:

$$d_{min} = h \cdot \tan(\theta) \quad \text{and} \quad d_{max} = h \cdot \tan(\theta + \phi) \tag{A.7}$$

and so the total forward distance projected onto the sensor is:

$$d_T = h \cdot [\tan(\theta + \phi/2) - \tan(\theta)] . \tag{A.8}$$

The smallest forward distance that must be detectable can be expressed similarly:

$$d_o = h \cdot [\tan(\theta + \phi_o) - \tan(\theta)] . \tag{A.9}$$

Following the guideline that this distance be one tenth of the body width, w, results in:

$$\frac{w}{10} = h \cdot [\tan(\theta + \phi_o) - \tan(\theta)] . \tag{A.10}$$

An expression for the angle ϕ_o associated with this distance can be found as follows:

$$\left[\frac{w}{10 \cdot h} + \tan(\theta)\right] = \tan(\theta + \phi_o)$$

$$\left[\frac{w}{10 \cdot h} + \tan(\theta)\right] = \frac{\tan(\theta) + \tan(\phi_o)}{1 - \tan(\theta) \cdot \tan(\phi_o)}$$

$$\left[\frac{w}{10 \cdot h} + \tan(\theta)\right] - \left[\frac{w}{10 \cdot h} + \tan(\theta)\right] \cdot \tan(\theta) \cdot \tan(\phi_o) = \tan(\theta) + \tan(\phi_o)$$

$$\tan(\phi_o) = \frac{\left[\frac{w}{10 \cdot h} + \tan(\theta)\right] - \tan(\theta)}{\left[\frac{w}{10 \cdot h} + \tan(\theta)\right] \cdot \tan(\theta) + 1} = \frac{\frac{w}{10 \cdot h}}{\left[\frac{w}{10 \cdot h} + \tan(\theta)\right] \cdot \tan(\theta) + 1} \tag{A.11}$$

From Fig. X (a), it can be seen that $\tan(\phi_o) = v_o / f$, which leads to:

$$v_o = \frac{f \cdot \frac{w}{10 \cdot h}}{(\tan(\theta))^2 + \frac{w}{10 \cdot h} \cdot \tan(\theta) + 1} . \tag{A.12}$$

Appendix B Expressing the Imaging Model in OpenGL

The simple imaging model presented in section 4.3.2 corresponds to the perspective projection used in OpenGL, specified with the command glFrustum, and the parameters shown in Fig. B.1. As can be seen from the figure, the size of the sensor can be specified using *left, right, top, bottom*, and the viewing volume defines the field-of-view. The focal length of the system can be specified using the parameter *near*. In this way a sensor with a 2mm width and a 3mm focal length can easily be specified through the user interface of the simulator. The API expresses everything in its own native 'units' which in this case were consistently treated as millimeters.

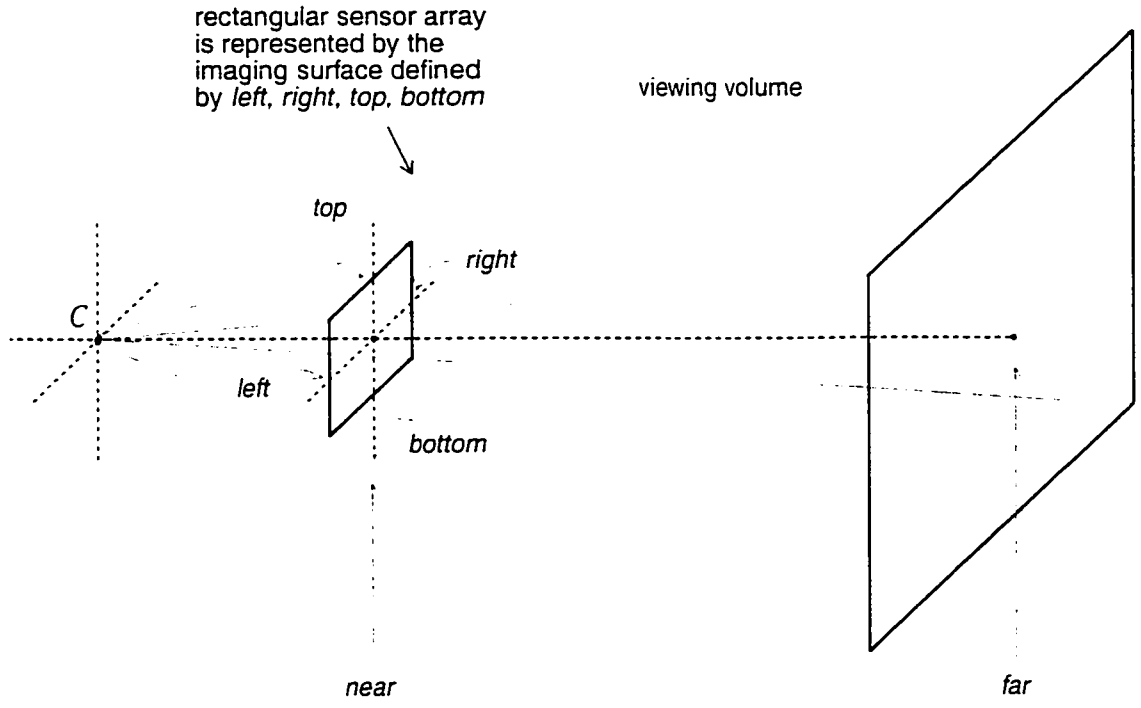


Fig. B.1 Notation used by the OpenGL interface to specify the imaging geometry.